S-100

Universal Hydrographic Data Model

Edition 5.1.0 – October 2023

IHO

International
Hydrographic
Organization

Published by the
International Hydrographic Organization
4b quai Antoine 1er
Principauté de Monaco
Tel: (377) 93.10.81.00
Fax: (377) 93.10.81.40
info@iho.int
www.iho.int

S-100 – Part 0

Overview

Page intentionally left blank

Contents

Page intentionally left blank

Foreword

Development of S-100 – the *IHO Universal Hydrographic Data Model* was included in the IHO Work Programme in 2001. S-100 has been developed by the IHO Transfer Standards Maintenance and Applications Development (TSMAD) Working Group with active participation from hydrographic offices, industry and academia. Since 2015, S-100 has been further developed by the S100 Working Group (S100WG).

S-100 provides a contemporary hydrographic geospatial data standard that can support a wide variety of hydrographic-related digital data sources, and is fully aligned with mainstream international geospatial standards, in particular the ISO 19100 series of geographic standards, thereby enabling the easier integration of hydrographic data and applications into geospatial solutions.

The primary goal for S-100 is to support a greater variety of hydrographic-related digital data sources, products, and customers. This includes the use of imagery and gridded data, enhanced metadata specifications, unlimited encoding formats and a more flexible maintenance regime. This enables the development of new applications that go beyond the scope of traditional hydrography - for example, high-density bathymetry, seafloor classification, marine GIS, et cetera. S-100 is designed to be extensible and future requirements such as 3-D, time-varying data (x, y, z, and time) and Web-based services for acquiring, processing, analysing, accessing, and presenting hydrographic data can be easily added when required.

The S-100 development and maintenance process is specifically aimed at allowing direct input from non-IHO stakeholders, thereby increasing the likelihood that these potential users will maximise their use of hydrographic data for their particular purposes.

S-100 will eventually replace S-57 – the established *IHO Transfer Standard for Digital Hydrographic Data*. Although S-57 has many good aspects, it has some limitations:

- S-57 has been used almost exclusively for encoding Electronic Navigational Charts (ENCs) for use in Electronic Chart Display and Information Systems (ECDIS).
- S-57 is not a contemporary standard that is widely accepted in the GIS domain.
- It has an inflexible maintenance regime. Freezing standards for lengthy periods is counter-productive.
- As presently structured, it cannot support future requirements (for example, gridded bathymetry, or time-varying information).
- Embedding the data model within the encapsulation that is, file format) restricts the flexibility and capability of using a wider range of transfer mechanisms.
- It is regarded by some as a limited standard focused exclusively for the production and exchange of ENC data.

The transition from S-57 to S-100 will be carefully monitored by the IHO to ensure that existing S-57 users, particularly ENC stakeholders, are not adversely affected. S-57 will continue to exist as the designated format for ENC data for the foreseeable future.

In the meantime, all existing and potential users of hydrographic information and data are encouraged to use S-100 as the basis for new applications, seeking input to the further development of the standard if their particular requirements are not yet catered for.

Document Control

Edition Number	Date	Reference
1.0.0	January 2010	IHO Circular Letter No 83/2009 4 December 2009
2.0.0	June 2015	IHO Circular Letter No 39/2015 05 June 2015
3.0.0	June 2017	IHO Circular Letter No 32/2017 02 May 2017
4.0.0	December 2018	IHO Circular Letter No 60/2018 17 December 2018
5.0.0	December 2022	IHO Circular Letter No 45/2022 12 December 2022
5.1.0	October 2023	IHO Circular Letter No xx/2023 xx October 2023

Introduction

Standards should encapsulate the use of best practice methods and procedures. They should include guidance on how to implement efficient production methods and optimize the quality of an organizations products and services, and should also enable interoperability between disparate technologies through the use of common interfaces. The S-100 standard attempts to achieve all of these objectives. Furthermore it provides a framework of components that can be used by interested communities to develop their own maritime geospatial products and services.

The S-100 standard has been developed with the advantage of hindsight based on experience gained through the development and use of the existing IHO Transfer Standard for Digital Hydrographic Data (known as S-57). S-100 has been documented using an object-oriented notation known as the Unified Modelling Language (UML). (Although UML defines nine types of diagrams, only class, object and package diagrams have been used in S-100).

The S-100 standard provides a theoretical framework of components that are based on the ISO 19100 series of standards and specifications. These standards and specifications are also used as the basis for most contemporary geospatial standards development activities and are closely aligned with other standards development initiatives such as the Open Geospatial Consortium (OGC).

The IHO has also developed an associated Geospatial Information (GI) Registry which can be used in conjunction with the S-100 standard. The IHO GI Registry contains the following additional components;

- Concept Register.
- Data Dictionary Register.
- Portrayal Register.
- Register of IHO Data Producer Codes.
- Register of S-100 based Product Specifications.
- A help and guidance repository containing supporting documentation and tools to support S-100 based Product Specification development, such as Catalogue Builders.

The IHO GI Registry provides the infrastructure and mechanisms required to manage and maintain the resources listed above, and to extend them as required.

NOTE S-100 provides a Schema and overarching management procedures for a Registry and its Registers and the IHO GI Registry is implemented using these concepts.

Page intentionally left blank

0-1 Scope

S-100 – IHO *Universal Hydrographic Data Model* comprises a set of related parts that give the user the appropriate tools and framework to develop and maintain hydrographic related data, products and registers. These standards specify, for hydrographic and related information, methods and tools for data management, processing, analysing, accessing, presenting and transferring such data in digital/electronic form between different users, systems and locations. By following this set of geospatial hydrographic standards users will be able to build constituent parts of an S-100 compliant product specification.

S-100 conforms as far as is reasonably possible to the ISO TC 211 series of geographical information standards, and where necessary has been tailored to suit hydrographic requirements.

S-100 details the standard to be used for the exchange of hydrographic and related geospatial data between national hydrographic offices as well as between other organizations and for its distribution to manufactures, mariners and other data users.

S-100 comprises multiple parts that profile standards developed by the ISO Technical Committee 211. ISO TC 211 is responsible for the ISO series of standards for geographic information. The objective is that, together, the standards will form a framework for the development of sector specific applications that use geographic information. S-100 is an example of such an application.

This standard specifies the procedures to be followed for:

1) establishing and maintaining registers of hydrographic and related information;
2) creating product specifications, feature catalogues and a definition of the general feature model;
3) using spatial, imagery and gridded data, and metadata specifically aimed at fulfilling hydrographic requirements.

0-2 Abbreviations used in this publication

2-D	Two-dimensional
2.5D	Two and a half dimensional
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
CRS	Coordinate Reference System
CSL	Conceptual schema language
DEF	Data Exchange Format
DIS	Draft International Standard
ECDIS	Electronic Chart Display and Information System
ECS	Electronic Chart System
ENC	Electronic Navigational Chart
EPSG	European Petroleum Survey Group
FCD	Feature Concept Dictionary
FDIS	Final Draft International Standard
GFM	General Feature Model
GML	Geography Markup Language
HDF	Hierarchical Data Format
HSSC	IHO Hydrographic Services and Standards Committee (formerly CHRIS)
IALA	International Association of Lighthouse Authorities
ICC	International Colour Consortium
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force
IHB	International Hydrographic Bureau

IHO	International Hydrographic Organization
IMO	International Maritime Organization
IOGP	International Association of Oil and Gas Producers (formerly OGP)
ISO	International Organization for Standardization
ISO/TC211	ISO Technical Committee for Geographic information/Geomatics
JPEG	Joint Photographic Experts Group
MRN	Maritime Resource Name
OCL	Object Constraint Language
ODP	Open Distributed Processing
OEM	Original Equipment Manufacturer
OGC	Open Geospatial Consortium
OMG	Object Management Group
OSI	Open Systems Interconnection
RENC	Regional ENC Coordinating Centre
RFC	Request for Comments
RNC	Raster Navigational Chart
RSS	Recommended Security Scheme
SENC	System-ENC
SKOS	Simple Knowledge Organization System
TC	Technical Committee
TIFF	Tagged Image File Format
TIN	Triangulated Irregular Network
TS	Technical Specification
TSMAD	Transfer Standard Maintenance and Application Development Working Group
S-100WG	S-100 Working Group
SVG	Scalable Vector Graphics
UML	Unified Modelling Language
URI	Uniform Resource Identifier
URL	Universal Resource Locator
XLink	XML Linking Language
XMI	XML Metamodel Interchange
XML	Extensible Markup Language
XSD	World Wide Web Consortium XML Schema Definition
XSL	eXtensible Stylesheet Language

0-3 Objectives of S-100

The objectives of S-100 are:

1) To comply with the emerging ISO standards for geographic information being produced by ISO TC 211;
2) To provide support for a greater variety of marine or hydrographic-related digital data, products and customers;
3) To separate the data content from the encoding format, enabling format neutral product specifications;
4) To enable manageable flexibility that can accommodate change. The intention is that product specifications will be allowed to evolve through extension without the need to publish new versions of existing product specifications;

5) To provide an ISO-conformant registry managed by the IHO containing registers such as feature concept dictionaries and product feature catalogues that are flexible and capable of managed expansion;
6) To provide separate registers for different user communities.

0-4 S-100 Parts

S-100 comprises multiple parts that are derived from various ISO 19100 series of standards.

Table 0-1 lists the individual parts, their associated part numbers and ISO 19100 conformance.

Table 0-1 — S-100 Parts

Part Title	Part Number	ISO19100 Standard
Conceptual Schema Language	S-100 Part 1	ISO 19103:2005, Geographic information - Conceptual schema language ISO
Management of IHO Geospatial Information Registers	S-100 Part 2	ISO 19135:2005, Geographic Information - Procedures for registration of items of geographic information
Concept and Data Dictionary Registers	S-100 Part 2a	ISO 19135:2005, Geographic Information - Procedures for registration of items of geographic information ISO 19126:2009, Geographic Information – Feature concept dictionaries and registers
Portrayal Register	S-100 Part 2b	ISO 19135:2005, Geographic Information - Procedures for registration of items of geographic information ISO 19126:2009, Geographic Information – Feature concept dictionaries and registers ISO 19117:2012, Geographic Information - Portrayal
General Feature Model and Rules for Application Schema	S-100 Part 3	ISO 19109:2005, Geographic information - Rules for application schema
Metadata	S-100 Part 4a	ISO 19115-1:2014, Geographic information – Metadata. Amended by Amendment 1, 2018
Metadata for Imagery and Gridded Data	S-100 Part 4b	ISO 19115-1:2014, Geographic information – Metadata – Part 1: Fundamentals. As amended by Amendment 1, 2018 19115-2:2009. Geographic information – Metadata – Part 2: Extensions for imagery and gridded data
Metadata – Data Quality	S-100 Part 4c	ISO 19113, Geographic information - Quality principles ISO 19114, Geographic information - Quality evaluation procedures ISO 19138, Geographic information - Quality measures
Feature Catalogue	S-100 Part 5	ISO 19110:2005, Geographic Information - Methodology for feature cataloguing
Coordinate Reference Systems	S-100 Part 6	ISO 19111:2007, Geographic information - Spatial referencing by coordinates
Spatial Schema	S-100 Part 7	ISO 19107:2003, Geographic information - Spatial schema
Imagery and Gridded Data	S-100 Part 8	ISO 19123:2007, Geographic information - Schema for coverage geometry and functions ISO 19129, Geographic information - Imagery, Gridded and Coverage Data Framework

Portrayal	S-100 Part 9	
Portrayal (Lua)	S-100 Part 9a	Lua Portrayal Implementation
Encoding Formats	S-100 Part 10	
ISO/IEC 8211 Encoding	S-100 Part 10a	ISO/IEC 8211:1994, Specification for a data descriptive file for information interchange structure implementations
GML Encoding	S-100 Part 10b	ISO 19136:2007 Geographic information - Geography Markup Language
HDF5 Encoding	S-100 Part 10c	HDF5 Data Model and File Format
Product Specifications	S-100 Part 11	ISO 19131:2008 Geographic information – Data product specifications
S-100 Maintenance Procedures	S-100 Part 12	
S-100 Scripting Language	S-100 Part 13	Provides scripting support for S-100 based Product Specifications
Online Communication Exchange	S-100 Part 14	Specifies an online exchange mechanism for S-100
Encryption and Data Protection	S-100 Part 15	Specifies encryption and data protection for S-100 based products
Interoperability Catalogue Model	S-100 Part 16	Defines a framework for creating rules for the interoperation of S-100 data products
Harmonised Portrayal of S-100 Products	S-100 Part 16a	Specifies the principles for harmonising portrayal and other presentational functionalities across different S-100 based data products
Discovery Metadata for Information Exchange Catalogues	S-100 Part 17	Provides a specification for describing and creating exchange catalogues that enables users to identify, discover and manage content of S-100 Exchange Sets
Language Packs	S-100 Part 18	Provides the generic methodology for implementing multi-lingual support; and informative examples for a primary use case, the creation of multi-lingual support for S-100 Feature Catalogues.

0-4.1 Profiles

The ISO base standards provide a large number of options to the developer wishing to use them for practical applications. The concept of a profile provides a method of adapting the base standards so that they meet specific implementation requirements.

A profile is a set of one or more base standards and, where applicable, the identification of chosen clauses, classes, subsets, options and parameters of those base standards, that are necessary to accomplish a particular function. ISO 19106 describes two levels of conformance for profiling the ISO 19100 series of standards. Each part of S-100 documents the level used in the conformance statement for that part.

S-100 is a set of profiles of the ISO TC 211 standards for Geographic Information. The relationship between S-100 standard core parts and their ISO base classes is shown in Table 0-1.

0-4.2 Part 1 – Conceptual Schema Language

This Part defines the conceptual schema language and basic data types for use within the IHO community. It identifies the combination of the Unified Modelling Language (UML) static structure diagram, and a set of basic data type definitions as the conceptual schema language for specification of geographic information.

0-4.3 Part 2 – Management of IHO Geospatial Information Registers

The International Hydrographic Organization (IHO) has developed a Registry in conformance with ISO 19135 - *Procedures for registration of items of geographic information*. This Registry contains an extensible number of Registers, encompassing Feature Concepts, Data Dictionaries, Portrayal and Meta Data. This Part describes the contents, structure and management of these Registers.

0-4.4 Part 2a – Concept and Data Dictionary Registers

The Concept Register specifies hydrographic core conceptual information (definitions, camelCase, etc) that may be used to describe geographic or meta data information. The use of a Register to store hydrographic definitions significantly improves the IHOs ability to manage and extend multiple products based on S-100 which can be made available for use in a relatively short timescale. As such, the Register supports wider use of registered items by making them publicly available; and increases their visibility to potential users. The Concept Register is the primary resource where all registered concepts are stored and managed as "stateless" concepts (that is, items are not assigned a type and there is no defined binding of concepts to other concepts within the Register). Each concept shall be included as a single instance in the Register and will be used as the common source from which Data Dictionary Register and Meta Data Register concepts are derived and used to model features, attributes etc. for use in S-100 based Product Specifications.

The Data Dictionary Register expands on the concepts stored in the Concept Register (S-100 Part 2a), by including the assignment of item types and feature binding in discrete Domains within the Register. This allows S-100 based Product Specification developers to develop their data models to best suit their specific requirements for representation of the real world.

This Part describes the content of the Registers and specifies procedures to be followed in establishing, maintaining, and publishing dictionaries of unique, unambiguous and permanent identifiers that are assigned to items of geographic, hydrographic and metadata information. In order to accomplish this purpose, this Part specifies elements of information that are necessary to provide identification and definitions to the registered items.

0-4.5 Part 2b – Portrayal Register

This Part describes the content of the portrayal register. A Portrayal Register specifies the portrayal of data. The portrayal of data is independent of the data but closely related to the data. That is the attributes within the data set drive the portrayal process, but there may be many different portrayals for the same data. The use of a Register to store aspects of portrayal will significantly improve the IHO's ability to manage and extend multiple products based on S-100 which can be made available for use in a relatively short timescale. This Register will support wider use of registered items by making them publicly available and increase their visibility to potential users.

0-4.6 Part 3 – General Feature Model

This Part introduces the rules for developing an application schema which is a fundamental element of any S-100 based product specification. Equally fundamental to the creation of the application schema is a General Feature Model (GFM) which is a conceptual model for features, their characteristics and associations. It also introduces the concept of the information type. The GFM is a profile of the GFM presented in ISO 19109 Rules for Application Schemas.

0-4.7 Part 4 – Metadata

Increasingly, hydrographic organizations are collecting, storing and archiving large quantities of digital data which are becoming an important national asset. Characterising the data resources and facilitating their discovery, access, retrieval, and use is required in order for users to be able to understand the assumptions and limitations of data resources and evaluate the resources' applicability for their intended use. Further, knowledge of the quality of hydrographic data is crucial for the application for the data, as different users and different applications often have different data quality requirements. In order to achieve this, data custodians will need to record information about the characteristics and quality of their data (that is metadata) in order to facilitate discovery, access, retrieval and use, and assure reliability.

ISO 19115-1, 19115-2, and 19157 provide an abstract structure for describing digital geographic information by defining the resources' characteristics and quality metadata elements and establishing a common set of metadata terminology, definitions, and extension procedures.

This Part also describes how to use ISO 19115-1, 19115-2 and 19157 metadata classes, elements and conditions, and incorporates rules for populating quality metadata. It also incorporates quality measures as described in ISO 19113, 19114 and 19157.

0-4.8 Part 5 – Feature Catalogue

A Feature Catalogue is a document that describes the content of a data product. It uses item types, for example, features and attributes, from one or more Feature Data Dictionaries. The basic level of classification in a Feature Catalogue is by feature type and information type. A Feature Catalogue should be available in electronic form for any set of geographic data that contains features. A Feature Catalogue may also comply with the specifications of this part of S-100 independently of any existing set of geographic data.

A Feature Catalogue is defined for each Product Specification. Features and attributes are bound in a Feature Catalogue. The definitions of features and attributes are drawn from a Feature Data Dictionary.

This Part defines the methodology for cataloguing feature types. It also specifies how the classification of feature types is organized into a Feature Catalogue and presented to the users of a set of geographic data. This Part is applicable to creating catalogues of feature types in previously un-catalogued domains and to revising existing Feature Catalogues to comply with standard practice. This Part applies to the cataloguing of feature types that are represented in digital form. Its principles can be extended to the cataloguing of other forms of geographic data.

Part 5 is applicable to the definition of geographic features at the type level. This international standard is not applicable to the representation of individual instances of each type.

0-4.9 Part 6 – Coordinate Reference Systems

This Part is applicable to producers and users of hydrographic information. Its principles can be extended to many other forms of geographic information such as maps, charts, and text documents.

This Part defines the conceptual schema for the description of spatial referencing by coordinates. It describes the minimum data required to define a one, two and three dimensional spatial coordinate reference. All the elements necessary to fully define spatial referencing by means of coordinate systems and datums are contained in this section. It also describes the information required to change coordinates from one coordinate reference system to another and all the elements necessary to describe the parameters and methods of coordinate operations. Coordinate operations include projections and datum transformations.

Coordinate reference system information can be presented in full using the elements defined in this part or by reference to a register of coordinate reference system information. A register of coordinate reference system information may be managed in accordance with ISO 19135 (see Part 2).

There are no plans for the IHO to implement a register of coordinate reference systems. An example of an existing register of coordinate reference system information which may be used is the EPSG geodetic parameter dataset which is managed by the Geodesy Subcommittee of the IOGP Geomatics Committee. Complete CRS definitions may be communicated by means of the namespace EPSG and a code, such as 4326 (that is, EPSG:4326). This code within the EPSG namespace identifies the ellipsoidal coordinate system based on WGS84 datum. The EPSG database is not managed in accordance with ISO 19135.

0-4.10 Part 7 – Spatial Schema

This Part defines the information necessary for describing and manipulating the spatial characteristics of features. It is based on ISO 19107 - *Geographical Information - Spatial schema*, however the spatial requirements of S-100 are less comprehensive than the requirements of ISO 19107. This profile contains the subset of ISO 19107 classes which are included in S-100.

0-4.11 Part 8 – Imagery and Gridded Data

This Part identifies the content model for gridded data for use in Hydrographic and related applications, including imagery and gridded data. It describes the organization, type of grid and associated metadata and spatial referencing. The encoding and portrayal of imagery and gridded data is external to this part of S-100, although the manner by which encoding and portrayal makes use of the identified content models are identified. This Part is based on the ISO 19129 Imagery, gridded and coverage data framework.

0-4.12 Part 9 – Portrayal

This Part specifies the portrayal model for defining and organizing symbols and portrayal rules necessary to portray S-100 product Features.

0-4.13 Part 9a – Portrayal (Lua)

This Part defines the additions and changes to S-100 Part 9 necessary to implement portrayal using the scripting mechanism defined in S-100 Part 13. Products which specify use of a portrayal catalogue as described in this part must also require implementation of S-100 Part 13.

0-4.14 Part 10 – Encoding Formats

This Part covers encoding formats. S-100 does not mandate particular encoding formats so it is left to developers of Product Specifications to decide on suitable encoding standards and to document their chosen format. The issue of encoding information is complicated by the range of encoding standards that are available. Table 0-2 provides an incomplete list of available encoding standards from which Schemas can be developed as extensions to S-100 as required.

Table 0-2 – Example Encoding Standards

Encoding Name	Description
ISO/IEC 8211	The encoding standard currently used to encode S-57 ENC data.
GML	Geography Markup Language
XML	Extensible Markup Language
GeoTIFF	Extension of the TIFF specification to allow the storage of geo-referencing information
HDF-5	Hierarchical Data Format version 5
JPEG2000	Joint Photographic Experts Group - Commonly used method for the compression of photographic images

Successful data interchange depends on knowledge of the content, defined in the Feature Catalogue, and the structure, defined in the Application Schema, of a dataset, and the encoding rules that are applied.

0-4.15 Part 10a – ISO/IEC 8211 Encoding Schema

This Part specifies the structure and physical constructs required for the implementation of exchange data sets encoded in the ISO 8211 format.

0-4.16 Part 10b – GML Encoding

This Part specifies the structure and physical constructs required for the implementation of the Geographic Markup Language data format.

0-4.17 Part 10c – HDF5 Data Model and File Format

This Part specifies the structure and constructs required for the implementation of exchange datasets encoded in the Hierarchical Data Format version 5 (HDF5).

0-4.18 Part 11 – Product Specifications

This Part explains Product Specifications. It is a descriptive IHO profile of ISO 19131 for data Product Specifications and describes data Product Specifications for hydrographic and hydrographically-related requirements for geographic data products.

The aim of this profile is to ensure a clear and consistent structure for any data Product Specification. This profile will conform with all the other standards that have been developed under the IHO S-100 framework.

A Product Specification is a description of all the features, attributes and relationships of a given application and their mapping to a dataset. It is a complete description of all the elements required to define a particular geographic data product.

0-4.19 Part 12 – Maintenance

This Part specifies procedures to be followed in maintaining and publishing the various Parts of S-100. It does not cover the maintenance of the S-100 Registry, as Register owners specify the procedures for updating their Registers. Additionally, it does not cover the maintenance regime of product specifications that are written in accordance to S-100.

NOTE All S-100 based Product Specifications will include a maintenance section.

0-4.20 Part 13 – Scripting

This Part defines a standard mechanism for including scripting support in S-100 based products. Scripting provides for processing of S-100 based datasets via script files written in the Lua programming language.

0-4.21 Part 14 – Online Communication Exchange

This Part describes the components and processes needed to specify an online exchange of information. It could be a set of data or data which may have a continuous nature. The latter is also known as “streaming data”, wherein the data requires a more dynamic information flow to be available; that is, beyond that found with the exchange of static datasets mostly handled as files.

0-4.22 Part 15 – Encryption and Data Protection

This Part specifies the mechanisms, structures and content required for the implementation of copy protections and/or authentication methods by S-100 product specifications. It defines standardized methods and algorithms for the encryption of file based components of datasets as well as feature and portrayal catalogues. Algorithms and methods for the production of digital signatures are defined as well as the surrounding infrastructure required for key management and identity assurance within the IHO Data Protection Scheme.

0-4.23 Part 16 – Interoperability Catalogue Model

This Part defines a framework for creating rules for the interoperation of S-100 data products, including harmonized graphical presentations and handling of alarms and indications. It can be used to establish system specific rules which are contained in an Interoperability Catalogue, a type of meta-product that describes how groups of products are to be used and displayed simultaneously.

0-4.24 Part 16a – Harmonised Portrayal of S-100 Products

This Part specifies the principles for harmonising portrayal and other presentational functionalities across different S-100 based data products for the purpose of improving the user experience and reducing ambiguities within systems utilising multiple S-100 based data products. It also describes the relevant International Maritime Organization (IMO) guidance and resources within International Hydrographic Organization (IHO) that support efforts in portrayal harmonisation. It does not address the portrayal process, functionality, or architecture, which are addressed in other S-100 Parts (especially Parts 9 and 9A), but instead focuses on presentational design aspects, such as display organisation, colours, and symbology.

0-4.25 Part 17 – Discovery Metadata for Information Exchange Catalogues

This Part provides a specification for describing and creating Exchange Catalogues that enables users to identify, discover and manage content of the S-100 Exchange Sets. More importantly it leverages XML to allow machine to machine discovery and exchange of information about geographic datasets commonly produced by hydrographic organizations. Its purpose is the creation of metadata records that provide information about the identification, spatial and temporal extent, quality, Application Schema, spatial reference system, and distribution of digital geographic data. It is applicable to the cataloguing of datasets, clearinghouse activities, and the full description of geographic and non-geographic resources.

For information exchange, there are several categories of metadata required: metadata about the overall Exchange Catalogue; metadata about each of the datasets contained in the Catalogue; and metadata about the support files that make up the package. If the Exchange Catalogue contains any Feature, Portrayal or Interoperability Catalogues there is a provision to carry additional metadata about those.

This Part is intended for developers and implementers of metadata applications, and provides a basic understanding of the principles and the overall requirements for standardisation of geographic information. It should be used in conjunction with the standards listed under clause S-100 Part 4a, clause 4a-4 – Normative references.

0-4.26 Part 18 – Language Packs

This Part details how multi-lingual support for XML elements of the S-100 framework may be implemented. A generic mechanism and structures are described for production of individual language packs which implement translations of any XML content.

This is designed to provide multi-lingual instances of XML resources which support Product Specifications for provision to end users. Implementing systems are then able to construct translated instances of those supporting resources. This Part is not specific to any one individual class of XML resource. It does not detail how multi-lingual support may be added to S-100 Product Specifications, datasets or any external resources they may reference. It provides a generic mechanism which can be applied to any XML based elements of the S-100 framework to adapt them for multi-lingual implementations.

This Part of S-100 provides the generic methodology for implementing such support; and informative examples for a primary use case, the creation of multi-lingual support for S-100 Feature Catalogues.

S-100 – Part 1

Conceptual Schema Language

Page intentionally left blank

Contents

Page intentionally left blank

1-1 Scope

This Part defines the Conceptual Schema language and basic data types for use within the IHO community. It identifies the combination of the Unified Modelling Language (UML) static structure diagram, and a set of basic data type definitions as the Conceptual Schema language for specification of geographic information. (UML is a standardized general-purpose modelling language in the field of software engineering. It includes a set of graphical notation techniques to create abstract models of specific systems. UML combines the best practice from data modelling concepts such as entity relationship diagrams, work flow, object modelling and component modelling).

Secondly, this Part provides guidelines on how UML should be used to create standardized geographic information and service models that are a basis for achieving the goal of interoperability. Since it deals with the UML, a section with specific UML terms and definitions is provided, in addition to these terms being included in S-100 Annex A (Terms and Definitions).

1-2 Conformance

Any Conceptual Schema written for a specification that claims conformance to this part of S-100 shall conform to the rules set out in clause 5. This profile conforms to conformance class 2 of ISO 19106:2004.

1-3 Normative references

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

ISO 19103:2005(E), *Geographic information — Conceptual schema language*

ISO 8601:2004(E), *Data elements and interchange formats — Information interchange — Representation of dates and times*

ISO 19136: *Geographic Information – Geography Markup Language*

ISO 25964-1: *Information and documentation — Thesauri and interoperability with other vocabularies — Part 1: Thesauri for information retrieval.*

ISO 25964-2: *Information and documentation — Thesauri and interoperability with other*

vocabularies — Part 2: Interoperability with other vocabularies

OGC 10-129r1: *Geographic Information – Geography Markup Language (GML) – Extended schemas and encoding rules*

OMG Unified Modelling Language (OMG UML), Superstructure, V2.1.2

RFC 3986, *Uniform Resource Identifier (URI): Generic Syntax*. T. Berners-Lee, R. Fielding, L. Masinter. Internet Standard 66, IETF. URL: http://www.ietf.org/rfc/rfc3986.txt or http://www.rfc-editor.org/info/std66

RFC 2141, *URN Syntax*. R. Moats. IETF RFC 2141, May 1997. URL: http://www.rfc-editor.org/info/rfc2141

RFC 8089, *The “file“ URI Scheme*, February 2017. URL: https://www.ietf.org/rfc/rfc8089.txt

SKOS: *SKOS – Simple Knowledge Organization System – Reference*. W3C Recommendation, 2009. http://www.w3.org/TR/2009/REC-skos-reference-20090818/

1-4 The S-100 UML Profile

1-4.1 Introduction

This clause provides rules and guidelines on the use of UML within the field of geographic information.

The sub-clauses are structured as follows:

1) General usage of UML
2) Classes
3) Attributes
4) Basic data types
5) Predefined derived types
6) Enumerated types
7) Codelist types
8) Relationships and associations
9) Stereotypes
10) Optional, conditional and mandatory – attributes and associations
11) Naming and name spaces
12) Notes
13) Packages
14) Documentation of models in S-100

1-4.2 General usage of UML

UML (The Unified Modelling Language) shall be used in a manner that is consistent with UML 2. Normative models shall use class diagrams and package diagrams. Other UML diagram-types may be used informatively. All normative models shall contain complete definitions of attributes, associations, and appropriate data type definitions.

1-4.3 Classes

A class is a description of a set of objects that share the same attributes, operations, methods, relationships, behaviour and constraints. A class represents a concept being modelled. Depending on the kind of model, the concept may be based on the real world (for a conceptual model), or it may be based on implementation between platform independent system concepts (for specification models) or platform specific system concepts (for implementation models).

A classifier is a generalization of a class that includes other class-like elements, such as data types, actors and components. A UML class has a name, a set of attributes, a set of operations and constraints. In S-100 operations are not used. A class may participate in associations.

A class according to the S-100 parts is viewed as a specification and not as an implementation.

The use of multiple inheritance shall be minimized, because it tends to increase model complexity.

An Abstract class is specified by having the class name in italics.

1-4.4 Attributes

UML notation for an attribute has the form:
$_{opt}$visibility$_{opt}$ name : $_{opt}$package ::$_{opt}$ $_{opt}$type$_{opt}$ $_{opt}$[multiplicity] $_{opt}$ $_{opt}$= initial value$_{opt}$ $_{opt}${property-string}$_{opt}$

An attribute must be unique within the context of a class and its supertypes, or else be a derived attribute, that is an attribute redefined from a supertype.

The visibility of attributes is shown by the symbols in Table 1-1. Protected and private visibility is normally not used in the standard specifications. The appropriate visibility symbols shall be used. The same visibility symbols are used for associations.

Table 1-1 — Visibility of Attributes

Symbol	Description
+	Public visibility
#	Protected visibility
-	Private visibility
/	Derived Attribute

All attributes must be typed and the type must exist, the constructed/defined types. A type must always be specified, there is no default type.

If no explicit multiplicity is given, it is assumed to be 1.

An attribute may define a default value, which is used when an object of that type is created. Default values are defined by explicit default values in the UML definition of the attribute.

The following properties can be used:

- readOnly – the value of the attribute cannot be changed and must be initialised.
- ordered – applies to attributes of a multiplicity of more than one in which the order of the elements is meaningful and must be maintained.

EXAMPLES + center: Point = (0,0) {readOnly}
+ origin: Point [0..1] // multiplicity 0..1 means that this is optional
+ controlPoints : Point [2..*] {ordered}

1-4.5 Basic data types

1-4.5.1 General considerations

The basic data types are grouped into two categories:

1) Primitive types: Fundamental types for representing values, for instance CharacterString, Integer, Boolean, Date, Time, etc.
2) Complex types: A combination of types, for instance a combination of measure types and units of measurement.

The repertoire of basic data types is described in the following sub-clauses.

S-100 data formats may represent values using appropriate built-in or standard types. For example, the ISO 8211 format (Part 10a) represents the values of all thematic feature attributes in strings instead of using the ISO 8211 signed integer, unsigned integer, or signed floating point representations for thematic attributes of S-100 type Integer or Real.

1-4.5.2 Primitive types

The following primitive types are supported in the S-100 UML Diagrams.

Table 1-2 — Data Types

Name	Description
Integer	A signed integer number, the representation of an integer is encapsulation and usage dependent. EXAMPLES `29`, `-65547`
PositiveInteger	An unsigned integer number greater than 0.
NonNegativeInteger	An unsigned integer number greater than or equal to 0.
Real	A signed real (floating point) number consisting of a mantissa and an exponent, the representation of a real is encapsulation and usage dependent. EXAMPLES `23.501`, `-1.234E-4`, `-23.0`
Boolean	A value representing binary logic. The value can be either true or false.
CharacterString	A CharacterString is an arbitrary-length sequence of characters including accents and special characters from repertoire of one of the adopted character sets.
Date	A date gives values for year, month and day according to the Gregorian Calendar. Character encoding of a date is a string which shall follow the calendar date format (complete representation, basic format) for date specified by ISO 8601. EXAMPLE `19980918` (YYYYMMDD) In XML formats, the XML Schema standard type should be used instead of the ISO 8601 basic representation (which is not a standard type in XML). EXAMPLE: `1998-09-18`
Time	A time is given by an hour, minute and second in the 24-hour clock system. Character encoding of a time shall be a complete representation of the basic format as defined in ISO 8601. Complete representation means that hours, minutes and seconds shall be used. Basic format means that separating characters are omitted. Time is preferably expressed as Universal Time Coordinated (UTC). EXAMPLE `183059Z` Time may be expressed as a Local Time with a given offset to UTC. EXAMPLE `183059+0100` Time may be expressed as a Local Time without a specified offset to UTC. EXAMPLE `183059` The complete representation of the time of 27 minutes and 46 seconds past 15 hours locally in Geneva (in winter one hour ahead of UTC), and in New York (in winter five hours behind UTC), together with the indication of the difference between the time scale of local time and UTC, are used as examples. Geneva: `152746+0100` New York: `152746-0500` The service hours for a service, that is available all year in an area where Daylight Saving Hour affects the offset to UTC could be expressed as Local Time without specified offset. Opening: `074500` Closing: `161500` In XML formats, the XML Schema standard type should be used instead of the ISO 8601 basic representation (which is not a standard type in XML). EXAMPLES: `18:30:59Z`; `18:30:59+01:00`; `18:30:59`
DateTime	A DateTime is a combination of a date and a time type. Character encoding of a DateTime shall follow ISO 8601 (see above). EXAMPLE: `19850412T101530` In XML formats, the XML Schema standard type should be used instead of the ISO 8601 basic representation (which is not a standard type in XML). EXAMPLES: `1985-04-12T10:15:30`; `1985-04-12T10:15:30+01:00`; `1985-04-12T10:15:30Z`

S100_TruncatedDate	A S100_TruncatedDate allows a date or partial date to be given. At least one of the following components must be present with omitted elements replaced by the equivalent number of hyphens as determined by the format. Components: YYYY Year integer between 0000 and 9999 MM Month integer between 01 – 12 (inclusive) DD Day integer between 01 and 28, 29, 30, or 31 (inclusive), consistent with year and month values if these are specified This type can be used to encode recurring instants (see Part 3, clause 3-8). EXAMPLE 1 (ISO 8211, HDF5): YYYYMMDD with the unspecified component(s) replaced with hyphens such that the length of the encoding is always 8 characters: `----1217` representing 17 December of any year EXAMPLE 2 (XML): The appropriate XML Schema type should be used: `--12-17` representing 17 December of any year (conforming to the XML type **gMonthDay**) Part 10b provides further details about encoding in GML datasets.

1-4.5.3 Complex types

1-4.5.3.1 UnlimitedInteger

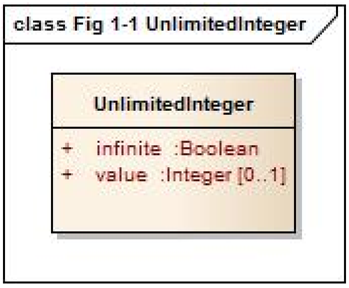


Figure 1-1 - UnlimitedInteger

A signed integer number whose value may be infinite.

1-4.5.3.2 Matrix

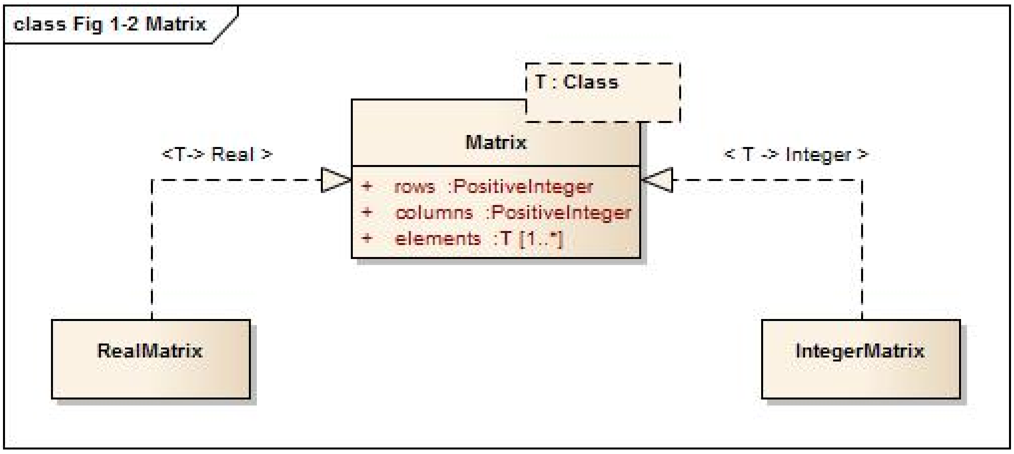


Figure 1-2 – Matrix

A grid of either real or integer elements.

1-4.5.3.3 S100_Multiplicity

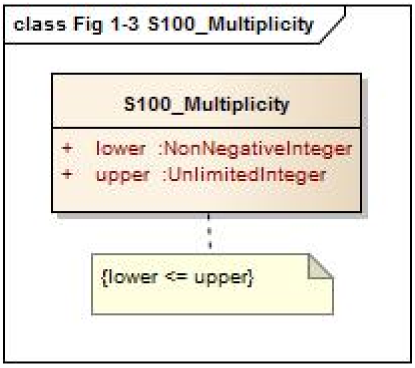


Figure 1-3 – S100_Multiplicity

Defines a multiplicity range from lower to upper. The upper boundary may be infinite.

1-4.5.3.4 S100_NumericRange

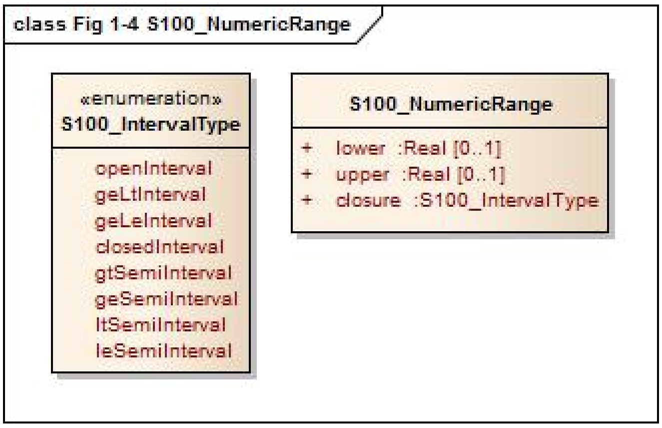


Figure 1-4 – S100_NumericRange

Specifies a numeric interval by its lower and upper boundary and the closure type of the interval.

NOTE The attribute **lower** must be used for all closures except **ltSemiInterval** or **leSemiInterval**. The attribute **upper** must be used for all closures except **gtSemiInterval** or **geSemiInterval**.

NOTE A single-value interval shall be encoded with **upper** = **lower** and set **closure** to **closedInterval**.

The closure of the interval is defined by the enumeration S100_IntervalType. The literals have the following meaning:

Table 1-3 — Interval Types

Name	Description	Notation	Definition
openInterval	The open interval	*(lower,upper)*	*lower < x < upper*
geLtInterval	The right half-open interval	*[lower,upper)*	*lower ≤ x < upper*
gtLeInterval	The left half-open interval	*(lower,upper]*	*lower < x ≤ upper*
closedInterval	The closed interval	*[lower,upper]*	*lower ≤ x ≤ upper*
gtSemiInterval	The left half-open ray	*(lower,∞)*	*lower < x*
geSemiInterval	The left closed ray	*[lower,∞)*	*lower ≤ x*
ltSemiInterval	The right half-open ray	*(-∞,upper)*	*x < upper*
leSemiInterval	The right closed ray	*(-∞,upper]*	*x ≤ upper*

NOTE Intervals using the round brackets (or) as in the general interval (lower,upper) or specific examples (-1,3) and (2,4) are called **open intervals** and the endpoints are not included in the set. Intervals using the square brackets [or] as in the general interval [lower,upper] or specific examples [-1,3] and [2,4] are called **closed**

intervals and the endpoints are included in the set. Intervals using both square and round brackets [and) or (and] as in the general intervals (lower,upper] and [lower,upper) or specific examples [-1,3) and (2,4] are called **half-closed intervals** or **half-open intervals**.

NOTE Intervals that have one of $\pm\infty$ as an end point are called rays or half-lines.

EXAMPLE The interval "(10,42)" indicates the set of all real numbers between 10 and 42 but does *not* include 10 or 42, the first and last numbers of the interval, respectively. The interval "[10,42]" includes every number between 10 and 42 *as well as* 10 and 42.

1-4.5.3.5 S100_UnitOfMeasure

A unit of measurement is a well defined comparator for a magnitude.

In S-100 a unit of measure is comprised of a name and optionally of a definition and a symbol.

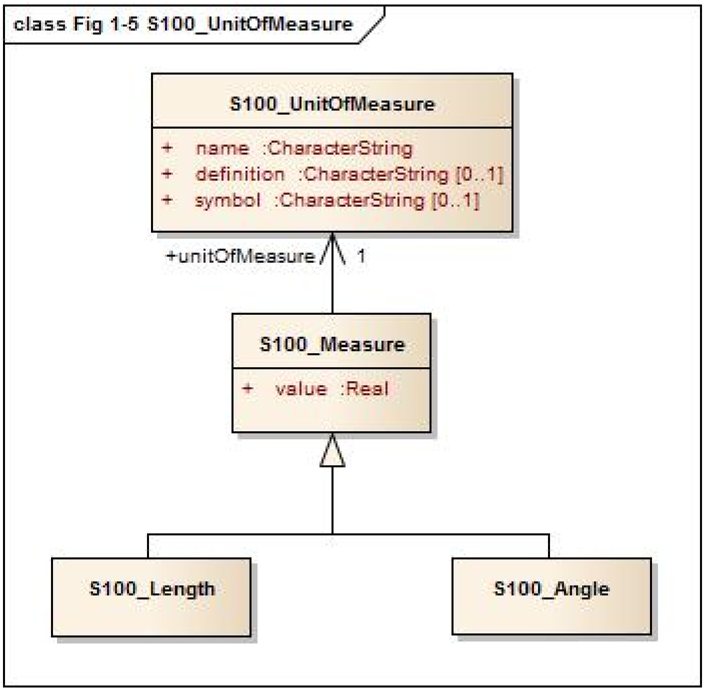


Figure 1-5 – S100_UnitOfMeasure

1-4.5.3.6 S100_Measure

A measure is the result of a measurement. A measurement is the estimation of the magnitude of some characteristic of an entity, such as its length or weight, relative to a unit of measurement. A measure consists of the actual magnitude (the value) and the unit of measurement.

1-4.5.3.7 S100_Length

The measure of distance as an integral, for example the length of curve, or the perimeter of a polygon as the length of the boundary.

1-4.5.3.8 S100_Angle

The amount of rotation needed to bring one line or plane into coincidence with another, generally measured in radians or degrees.

1-4.5.3.9 S100_IndeterminateDate

An indeterminate instant is an instant related by a specified temporal relation to a date specified in truncated format. The temporal relations allowed are ‘before’ and ‘after’ and indicate respectively that the instant is before or after the time instant specified by the date-time component.

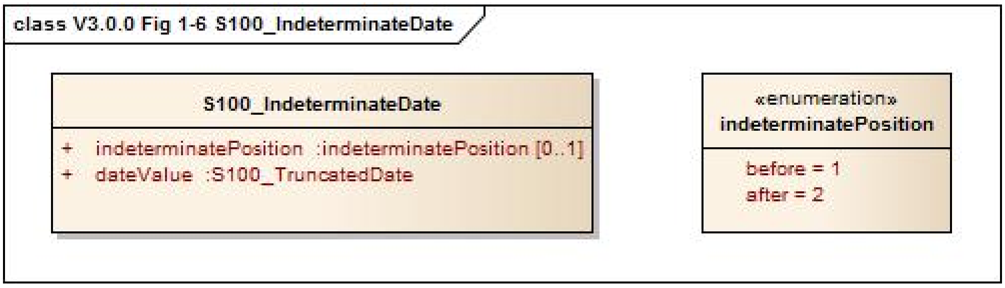


Figure 1-6 – S100_IndeterminateDate

Example (Informative): A mariner report dated at an unknown instant before the year 1950 is dated by an attribute *reportDate* with sub-attributes shown below:

Sub-attribute	Value	Remark
indeterminatePosition	1 (before)	At an indeterminate time before January 1, 1950.
value	1950----	

1-4.5.3.10 S100_TM_Instant

S100_TM_Instant represents the ISO19108 concepts of temporal instant. A temporal instant is a 0-dimensional geometric primitive representing position in time [ISO 19108:2002].

Role Name	Name	Description	Multiplicity	Data Type	Remarks
Class	S100_TM_Instant	A point in time. Multiple points may be specified as truncated dates used to represent recurring instants	--		One of *date*, *time* or *dateTime* must be populated
Attribute	date	A date or truncated date (see Table 1-2)	0..1	S100_TruncatedDate	
Attribute	time	A Time (see Table 1-2)	0..1	Time	
Attribute	dateTime	A DateTime (See table 1-2)	0..1	DateTime	

1-4.5.3.11 S100_TM_Period

S100_TM_Period represents the ISO19108 concepts of temporal period. A temporal period is a one-dimensional geometric primitive representing extent in time. [ISO19108:2002].

Role Name	Name	Description	Multiplicity	Data Type	Remarks
Class	S100_TM_Period	An extent in time.	--		Single value intervals are encoded with *begin* = *end* and *closure* = *closedInterval* (or omitted)
Attribute	closure	An S100_*IntervalType*. Default is *closedInterval*	0..1	S100_IntervalType	
Attribute	begin	Start of the period	0..1	S100_TM_Instant	
Attribute	end	End of the period	0..1	S100_TM_Instant	

1-4.6 Predefined derived types

Derived types are derived from the basic types or other derived types by restriction of the range of allowed values. The following derived types are defined in S-100. Product Specifications may define additional derived types.

Table 1-4 — Predefined Derived Types

Name	Description	Derived From
URI	A uniform resource identifier as defined in RFC 3986. Character encoding of a URI shall follow the syntax rules defined in RFC 3986. EXAMPLE http://registry.iho.int	CharacterString
URL	A uniform resource locator (URL) is a URI that provides a means of locating the resource by describing its primary access mechanism (RFC 3986). EXAMPLE http://registry.iho.int	URI
URN	A persistent, location-independent, resource identifier that follows the syntax and semantics for URNs specified in RFC 2141. EXAMPLE urn:iho:s101:1:0:0:AnchorageArea	URI

Attributes holding references to support files should be of URI attribute types and comply with the syntax in RFC 8089 for how to construct file references.

EXAMPLE The minimal representation of a local file with no authority field and an absolute path that begins with a slash "/".

* "file:/path/to/file"

NOTE In the context of Exchange Sets, datasets may reference a support file with an attribute value like file:/CABLES01.TXT which can be interpreted as

<ExchangeSetRoot>/SUPPORT_FILES/CABLES01.TXT

and as

/root/installation/folder/some/thing/else/support/files/folder/CABLES01.TXT

on the end-user installation.

1-4.7 Enumerated types

An enumerated type declaration defines a list of valid identifiers of mnemonic words. Attributes of an enumerated type can only take values from this list.

EXAMPLE

Figure 1-7 — Enumeration

Enumerations are modelled as classes that are stereotyped as <<enumeration>>. An enumeration class can only contain simple attributes which represent the enumeration values. Other information within an enumeration class is void. An enumeration is a user-definable data type, whose instances form a list of named literal values. Usually, both the enumeration name and its literal values are declared. The extension of an enumeration type will imply a Schema modification.

1-4.8 Codelist types

Codelist types may be used for open enumerations whose membership cannot be known at the level of the Product Specification, for reuse of information model fragments, or for more efficient Catalogue management. Specifically, they may be used:

a) for enumerations whose members are not all knowable at the level of the Application Schema;
b) for lists defined or controlled by external authorities;
c) for lists common to multiple S-100 domains;
d) if the set of allowed values needs to be extended without a major revision of the data specification;
e) long lists of potential values which would clutter or bloat Feature Catalogues.

For example, ISO 19115 (Metadata) defines several codelists, because it needs to define enumerated types whose membership is determined by domain and circumstances (for example distribution media).

A codelist type declaration must be one of the following 3 types:

1) An **open enumeration**, which is a list of valid key-value combinations (that is code-value mappings) with a provision for allowing user communities to provide allowed values in a specified format.
2) A **closed dictionary**, which is a dictionary (vocabulary) of key-value combinations in a known format, identifiable by a Uniform Resource Identifier and which can be located by the application of standard modern techniques for locating resources. Additional values cannot be provided.
3) An **open dictionary**, which is a dictionary (vocabulary) of key-value combinations in a known format, identifiable by a Uniform Resource Identifier, as defined above, with the additional proviso that additional values conforming to a specified format may be provided.

Codelists are modelled as classes that are stereotyped as <<S100_Codelist>>. Codelists of the first type must list the known literals as attributes. In the second and third types, no attributes are listed but the vocabulary is identified by a URI. A Codelist classifier must have tagged values which define its representation, extensibility, and anticipated encoding. Figure 1-8 shows 3 examples of codelists:

1) The **VerticalDatum** codelist is an example of a codelist modelled as an extensible enumeration (indicated by the tagged value *codelistType="open enumeration"*) which can be extended by values of the form "other: ...", indicated by the tagged value *encoding="other: [something]".*
2) The **ENCProducerCodes** codelist is an example of a codelist modelled by an external dictionary which can take only the values in that dictionary (indicated by tagged value *codelistType="closed dictionary"*). The dictionary is identified by the tagged value *URI=http://www.iho.int/producers/enc/ver1_5*.
3) The **Agency** codelist is an example of a codelist modelled by an external dictionary which can take additional values (indicated by the tagged value *codelistType="open dictionary"*). The dictionary is identified by the tagged value *URI=http://www.iho.int/agency/ver1_5.* The list can be extended by values of the form "other: ...", indicated by the tagged value *encoding="other: [something]".*

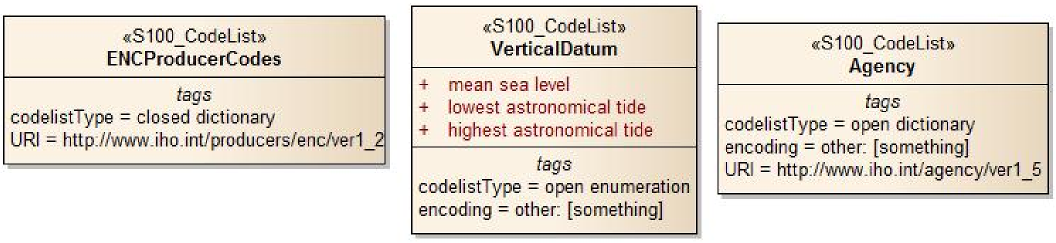


Figure 1-8 – Codelists

Implementations (and specific encodings) are allowed to depart from *encoding* hints. Different implementations may use different encoding schemes (and translation tables to other encoding schemes). For example preparation of a Feature Catalogue for an ISO 8211 encoding may transform a dictionary into an XML fragment which is merged into (or *Xinclude*'d in) the XML Feature Catalogue (obviously an additional procedure is needed for maintenance). This allows XML/GML encodings to use the dictionary while still allowing other encodings to function within their limitations.

1-4.9 Relationships and associations

1-4.9.1 Relationships

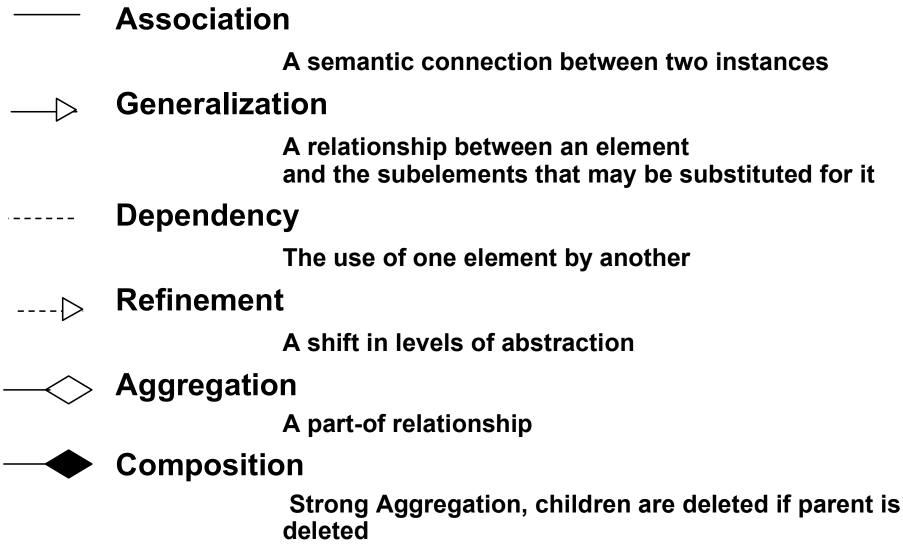


Figure 1-9 — Different kinds of relationships

A relationship in UML is a concrete semantic connection among model elements. Kinds of relationships include association, generalization, aggregation/composition, meta relationship, flow, and several kinds grouped under dependency. In ISO 19103 there is a clear distinction between the general term "relationship," and the more specific term "association". Both are defined for class to class linkages, but association is reserved for those relationships that are in reality instance to instance linkages. "Generalization," "realization" and "dependency" are class to class relationships. "Aggregation," and other object to object relationships, are more restrictively called "associations." It is always appropriate to use the most restrictive term in any case, so in speaking of instantiable relationships, use the term "association."

In S-100, generalization, dependency and refinement are used according to the standard UML notation and usage. In the following the usage of association, aggregation and composition is described further.

1-4.9.2 Association, composition and aggregation

An association in UML is the semantic relationship between two or more classifiers (for example class, interface, type, ...) that involves connections among their instances.

An association is used to describe a relationship between two or more classes. In addition to an ordinary association, UML defines two special types of associations called aggregation and composition. The three types have different semantics. An ordinary association shall be used to represent a general relationship between two classes. The aggregation and composition associations shall be used to create part-whole relationships between two classes.

A binary association has a name and two association-ends. An association-end has a role name, a multiplicity statement, and an optional aggregation symbol. An association-end shall always be connected to a class.

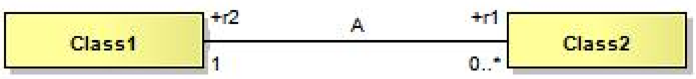


Figure 1-10 — Association

Figure 1-10 shows an association named "A" with its two respective association-ends. The role name is used to identify the end of an association, the role name r1 identifies the association-end which is connected to the class named class2. The multiplicity of an association-end can be one of exactly-one (1), zero-or-one (0..1), one-or-more (1..*), zero-or-more (0..*) or an interval (n..m). Viewed from the class, the role name of the opposite association-end identifies the role of the target class. We say that class2 has an association to class1 that is identified by the role r2 and which as a multiplicity of exactly one. The other way around, we can say that class1 has an association to class2 that is identified by the role name r1 with multiplicity of zero-or-more. In the instance model we say that class1 objects have a reference to zero-or-more class2 objects and that class2 objects have a reference to exactly one class1 object.

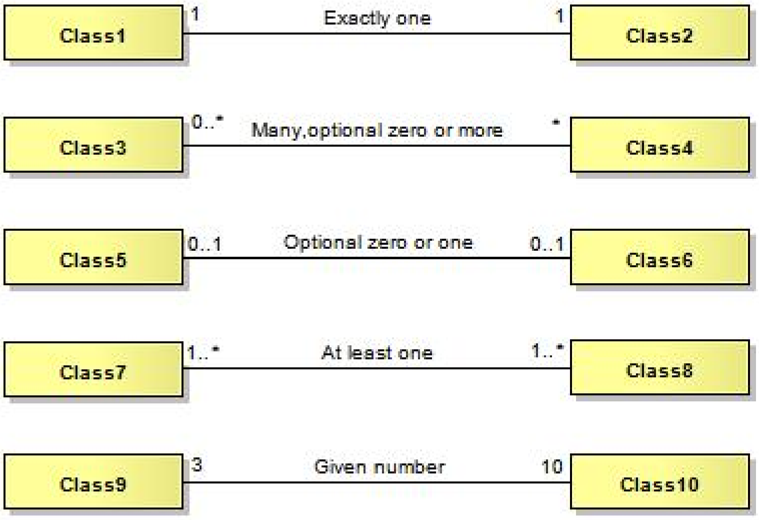


Figure 1-11 — Specification of multiplicity

The number of instances that can participate at one end in an association (or attribute) is specified in Figure 1-11.

An aggregation association is a relationship between two classes, in which one of the classes plays the role of container and the other plays the role of a containee. Figure 1-12 shows an example of an aggregation. The diamond-shaped aggregation symbol at the association-end close to class1 indicates that class1 is an aggregation consisting of class3. The meaning of this is that class3 is a part of class1. In the instance model, **class1** objects will contain one-or-more **class3** objects. The aggregation association shall be used when the containee objects (that represent the parts of a container object) can exist without the container object. Aggregation is a symbolic short-form for the part-of association but does not have explicit semantics. It allows for sharing of the same objects in multiple aggregations. If a stronger aggregation semantics is required, composition shall be used as described below. It is possible also to define role name and multiplicity at the diamond shaped end as well.

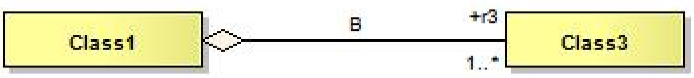


Figure 1-12 — Aggregation

A composition association is a strong aggregation. In a composition association, if a container object is deleted then all of its containee objects are deleted as well. The composition association shall be used when the objects representing the parts of a container object, cannot exist without the container object. Figure 1-13 shows a composition association in which the diamond-shaped composition symbol has a solid fill. Here **class1** objects consist of one-or-more **class4** objects, and the **class4** objects cannot exist unless the **class1** object also exists. The required (implied) multiplicity for the owner class is always one. The containees, or parts, cannot be shared among multiple owners.

It is possible also to define role name at the diamond shaped end as well, but the multiplicity will always be at most one. Composition shall be used to have the semantic effect of containment. Composition should be used with care, in particular one should consider the different requirements from various application perspectives before introducing this constraint. The application of the composition construct should be considered within the context of a model, (rather than the scope), where context means the application domain within which the application must be consistent. This is in order to prevent problems where different applications have different requirements for composition.

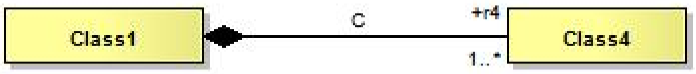


Figure 1-13 — Composition (strong aggregation)

All associations shall have cardinalities defined for both association ends. At least one role name shall be defined. If only one role name is defined, the other will by default be inv_rolename.

All association ends (roles) representing the direction of a relationship must be named or else the association itself must be named. The name of an association end (the rolename) must be unique within the context of a class and its supertypes. The direction of an association must be specified. If the direction is not specified, it is assumed to be a two-way association. If one-way associations are intended, the direction of the association can be marked by an arrow at the end of the line. If only the association is named, the direction of the association shall be specified.

Every UML association has navigability attributes that indicate which player in the association has direct access to the association opposite role. The default logic for an unmarked association is that it is two-way. Associations that do not indicate navigability are two-way in that both participants have equal access to the opposite role. Two-way navigation is not common or necessary in many client-to-server operations. The counterexample to this may be notification services, where the server often instigates communication on a prescribed event. The use of two-way relations that introduce unreasonable package dependencies shall be minimized. One-way relations shall be used when that is all that is needed.

If an association is navigable in a particular direction, the model shall supply a “role name” that is appropriate for the role of the target object in relation to the source object. Thus in a 2-way association, two role names will be supplied. The default role name is “the<target class name>” in which the target class is referenced from the source class (this is the default name in many UML tools). Association names are of secondary importance and actually are more for documentation purposes. Sometimes they can, however, be used for generating association-manager objects in environments that support associations as a first-class citizen concept.

Multiplicity refers to the number of relationships of a particular kind that an object can be involved in. If an association end were not navigable, putting a multiplicity constraint on it would require an implementation to track the use of association by other objects (or to be able to acquire the multiplicity through query). If this is important to the model, the association shall be two-way navigable to make enforcement of the constraint more tenable. In other words, a one-way relation implies a certain “don’t care” attitude towards the non-navigable end.

N-ary relationships, for N > 2 shall be avoided whenever possible, in order to reduce complexity. Multiplicity for associations are specified as UML multiplicity specifications. An association with role names can be viewed as similar to defining attributes for the two classes involved, with the additional constraint that updates and deletions are consistently handled for both sides. For one-way associations, it thus becomes equivalent to an attribute definition. The recommendation for S-100 is to use the association notation for all cases except for those involving attributes of basic data types.

1-4.10 Stereotypes

1-4.10.1 Use of standard UML stereotypes for class/classifier

In S-100 the following stereotypes are used:

a) <<Interface>> a definition of a set of operations that is supported by objects having this interface.
b) <<Type>> a stereotyped class used for specification of a domain of instances (objects), together with the operations applicable to the objects. A type may have attributes and associations.
c) <<Enumeration>> A data type whose instances form a list of named literal values. Both the enumeration name and its literal values are declared. Enumeration means a short list of well-understood potential values within a class. Classic examples are Boolean that has only 2 (or

3) potential values TRUE, FALSE (and NULL). Most enumerations will be encoded as a sequential set of Integers, unless specified otherwise. The actual encoding is normally only of use to the programming language compilers. In S-100 Codelists taken from the ISO 19100 standards are classified as enumerations.

d) <<MetaClass>> A class whose instances are classes. Metaclasses are typically used in the construction of metamodels. The meaning of metaclass is an object class whose primary purpose is to hold metadata about another class. For example, “FeatureType” and “AttributeType” are metaclasses for “Feature” and “Attribute”.'
e) <<DataType>> A descriptor of a set of values that lack identity (independent existence and the possibility of side effects). Data types include primitive predefined types and user-definable types. A DataType is thus a class with few or no operations whose primary purpose is to hold the abstract state of another class for transmittal, storage, encoding or persistent storage.
f) <<Codelist>> A data type whose instances form a list of named literals, some or all of whose members may not be known. The **Codelist** name is declared in the Application Schema. The list members may be described by either (i) a list of codes and corresponding literals augmented with a pattern allowing additional values conforming to a certain format, or (ii) a pointer to a resource consisting of a list of code/literal mappings. The resource is called a vocabulary or dictionary. Tagged values attached to the **Codelist** declaration indicate which form is used and the location of the resource (generally as a URI). Codelists should be used only when an enumeration is either unusable or inefficient (for example, if the full list of values is not known to the specification authors or the list of allowed values is long, volatile, controlled by another authority, and/or shared by multiple domains).

1-4.11 Optional, conditional and mandatory – attributes and associations

In UML all attributes are per default mandatory. The possibility to show multiplicity for attributes and association role names provide a way of describing optional and conditional attributes.

The default is mandatory which thus do not need to be specified. Where a multiplicity of 0..1 or 0..* is specified it means that this attribute may be present or may be omitted. A conditional attribute shall be shown as an optional attribute with a constraint statement in OCL. The condition shall be expressed as an OCL constraint in connection with the class declaration. This means that a null value must be represented in the instance model, for example a place holder element or a null value. An optional or conditional attribute shall never have a default value defined.

An attribute may be defined as conditional, meaning that it is optional depending on other attributes. The dependencies may be by existence-dependence of other (optional) attributes or by the values of other attributes. A conditional attribute is shown as optional with a conditional expression attached. The condition shall be written in a note directly associated with the attribute, or with the class and the name of the attribute on the first line. A conditional attribute shall never have a default value defined.

If unspecified, the default multiplicity for associations is 0..*, and the default multiplicity for attributes is 1.

1-4.12 Naming and name spaces

All classes shall have unique names. All classes shall be defined within a package. Class names shall start with an upper case letter. A class shall not have a name that is based on its external usage, since this may limit reuse. A class name shall not contain spaces. Separate words in a class name shall be concatenated. Each subword in a name shall begin with a capital letter, such as “XnnnYmmm”.

To ensure global uniqueness of class names, all class names shall be defined with bi-alpha prefixes. Bialpha prefixes allows for the use of _ after, such as in GM_Object. The geometry model uses bialpha prefixes (GM and TP). Other prefixes should be defined for other areas.

The name of an association must be unique within the context of a class and its supertypes or else it must be derived.

Attribute names shall start with a lower-case letter.

Example: firstName, lastName.

Precise technical names should be used for attributes and operations to avoid confusion.
Example: alphaCodeIdentifier, dateOfLastChange

Documentation fields should be used extensively to describe element.

Don't reiterate class names inside the attribute names. Keep names short if possible.
Example: class S-100_WorkingGroup, attribute workingGroupName.

Naming conventions are used for a variety of reasons, mainly readability, consistency and as a protection against case-sensitive binding.

The names of UML elements should:

1) Use precise and understandable technical names for classes, attributes.
   Example: index not i

2) For attributes and association roles capitalize only the first letter of each word after the first word that is combined in a name. Capitalize the first letter of the first word for each name of a class, package, type-specification and association names.
   Example: computePartialDerivatives (not computepartialderivatives or COMPUTEPARTIALDERIVATIVES)
   Example: CoordinateTransformation (not coordinateTransformation)

3) Keep names as short as practical. Use standard abbreviations if understandable, skip prepositions, and drop verbs when they do not significantly add to meaning of the name.
   - numSegment instead of numberOfSegments
   - Equals instead of IsEqual
   - value() instead of getValue()
   - initObject instead of initializeObject
   - length() instead of computeLength()

The UML naming scope with package::package::className allows for the same className to be defined in different packages. However, many UML tools do not currently allow for this. Therefore, a more restrictive naming convention is adopted:

1) Although the model is case sensitive, all class name should be unique in a case insensitive manner.

2) Class name should be unique across the entire model (so as not to create a problem with many UML tools).

3) Package names should be unique across the entire model (for the same reason).

4) Every effort should be applied to eliminate multiple classes instantiating the same concept.

1-4.13 Notes

Note boxes are used to comment on the model in general or on a specific item (that is class or association) of the model.

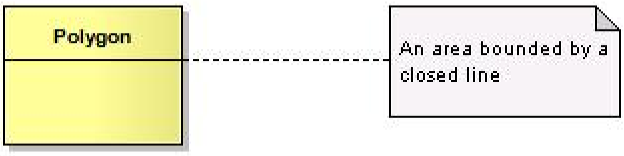


Figure 1-14 — Example note

1-4.14 Packages

A UML package is a container that is used to group declarations of subpackages, classes and their associations. The package structure in UML enables a hierarchical structure of subpackages, class declarations, and associations. A package shall be used to represent a Schema.

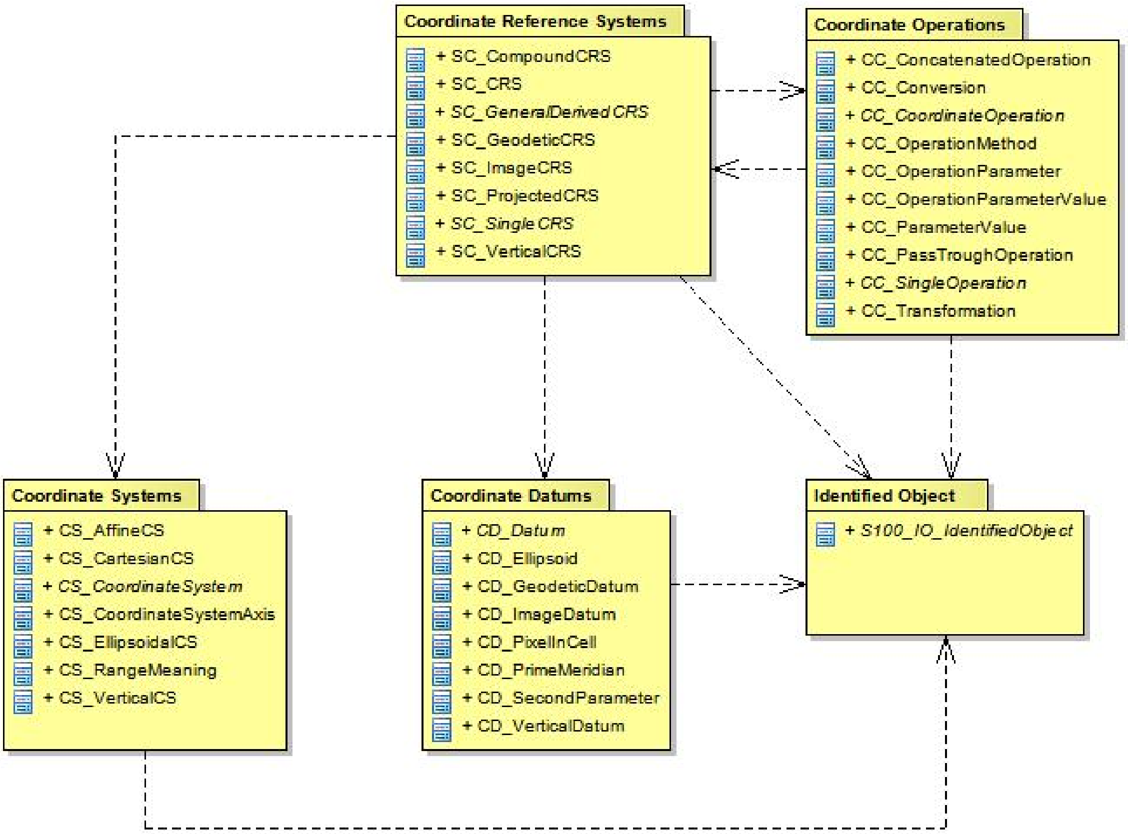


Figure 1-15 — Example package structure

The packages, classes and attributes in the Schema model can be identified by a qualified name. The form of the qualified names is *name1 : :name2 : :name3*, where *name1* is the name of the outermost package, *name2* is a name which appears within the namespace of *name1*, and *name3* is a name that appears within the namespace of *name2*. The standard UML ": :" symbol shall be used as a name separator. There is no limit of the depth of this namespace hierarchy.

EXAMPLE In the Spatial Schema there is a subpackage named Geometry which defines a class named GM_Object. This class has an association with role name SRS (Spatial Reference System). The fully qualified name for this association is: Spatial.Geometry : :GM_Object.SRS.

1-4.15 Documentation of models in S-100

In addition to the diagrams, it is necessary to document the semantics of the model. The meaning of attributes, associations, operations and constraints needs to be explained. This is done by means of context tables. A context table is defined for each class; it has the following columns:

- Role Name
- Name
- Description
- Multiplicity
- Data Type
- Remarks

The Role Name column specifies what property of the class is described in this row. Possible values are:

- Class – The class itself
- Attribute – An attribute of that class
- Association – An association to another class
- Enumeration – An enumerated data type
- Literal – A value of an enumerated data type

The Name column contains the name of the property. For association this is the role name used for the given class. In the Description column the semantics of the property are given. The Multiplicity column contains the number of occurrences of the property in the class. This also describes which properties are mandatory and which are optional. The Data Type column contains the name of the data type of the property. In the Remarks column additional information about the property can be expressed. This includes constraints or conditions. For the documentation of enumerated types the Multiplicity and Data Type column are not used.

The following example illustrates the use of context tables:

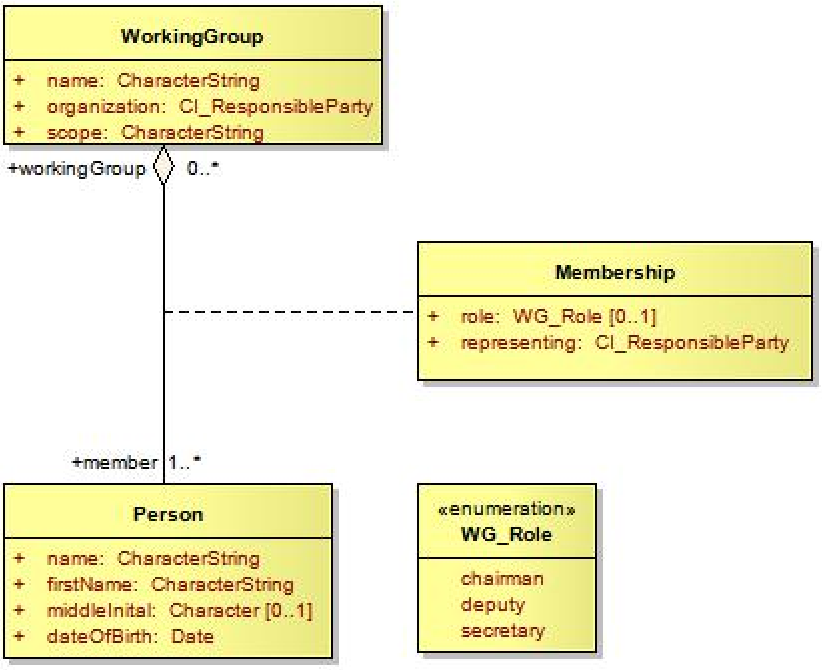


Figure 1-16 — Use of context tables

Role Name	Name	Description	Multiplicity	Data Type	Remarks
Class	WorkingGroup	A group of experts doing some useful work	-	-	
Attribute	name	The name of the working group	1	CharacterString	
Attribute	organization	The organization responsible for the working group	1	CI_ResponsibleParty	
Attribute	scope	The reason why so many people travel around the world	1	CharacterString	
Association	member	A person that is designated to contribute to the group	1..*	Person	

Role Name	Name	Description	Multiplicity	Data Type	Remarks
Class	Person	A human being	-	-	
Attribute	name	The name of the person	1	CharacterString	
Attribute	firstName	The first name of the person	1	CharacterString	
Attribute	middleInitial	The middle initial of the person	0..1	Character	
Attribute	dateOfBirth	The date when the person was born	1	Date	

Association	workingGroup	A working group the person contributes to	0..*	WorkingGroup	

Role Name	Name	Description	Multiplicity	Data Type	Remarks
Class	Membership	A class describing the membership of a person in a working group	-	-	
Attribute	role	The role that the person has in the working group	0..1	WG_Role	Ordinary member have no role
Attribute	representing	The organization which is represented by the person in the working group	1	CI_ResponsibleParty	

Role Name	Name	Description	Remarks
Enumeration	WG_Role	The roles people can have in a working group	
Literal	chairman	The gov'nor	
Literal	deputy	His best friend	
Literal	secretary	Poor man (or woman) has to have his (or her) fingers always on the keyboard	

S-100 – Part 2

Management of Registers

Page intentionally left blank

Contents

Page intentionally left blank

2-1 Scope

This Part of S-100 specifies procedures to be followed in maintaining and publishing Registers of unique, unambiguous and permanent identifiers that are assigned to items of geographic, hydrographic and metadata information. In order to accomplish this purpose, this Part describes the roles and responsibilities for the management of a Registry and its Registers. Specific administrative details of the IHO Geospatial Information (GI) Registry and its Registers is documented in IHO Publication S-99.

2-2 Conformance

This profile conforms to level 2 of ISO 19106:2004. The following is a brief description of the specializations and generalizations where the profile differs from ISO 19135:2005.

1) S100_RE_Register constrains the use of the attribute alternativeLanguages.
2) S100_RE_RegisterItem constrains the use of the attributes fieldOfApplication and alternativeExpression.
3) S100_RE_RegisterItem renames the attribute description to remarks.
4) S100_RE_ManagementInfo is a new class which amalgamates the classes RE_DecisionStatus, S100_RE_ProposalType, S100_RE_SubmittingOrganization, S100_RE_ItemStatus and RE_Disposition.
5) S100_RE_ProposalType is a new class which amalgamates the ISO 19135 classes RE_AdditionInformation, RE_ClarificationInformation, RE_AmendmentInformation and RE_AmendmentType.

2-3 Normative references

ISO 19126:2009, *Geographic Information – Feature concept dictionaries and registers*

ISO 19135:2005, *Geographic Information – Procedures for registration of items of geographic information*

ISO 8601:2004, *Data elements and interchange formats - Information interchange – Representation of dates and times*

IHO S-99, *Operational Procedures for the Organization and Management of the IHO Geospatial Information Registry*

2-4 General concepts

2-4.1 Registry

A Registry is the information system on which a Register or series of Registers is maintained.

2-4.1.1 Registry Owner

A Registry Owner has the authority to host the Registers and establish the policy for access. The Registry Owner decides whether a proposed Register shall be hosted on the Registry.

2-4.1.2 Registry Manager

The Registry Manager is responsible for the day-to-day operation of the Registry. This includes:

1) providing Registry access for Register Managers, Control Bodies, and Register Users;
2) ensuring that information about items in the Registers is readily available to users in relation to those items that have the status of valid, invalid, clarified, superseded, retired or processing;
3) accepting proposals and forwarding them to all Register Managers; and
4) managing the resolution of persistent URI identifiers to appropriate resources, but only if resolution services are provided on a Registry server.

2-4.2 Register

A Register is simply a managed list. It is easier to maintain than a fixed document, because new items can be added as needed to the Register; and existing items in the Register can be clarified, superseded or retired. Each Register item has one or more dates associated with it that indicate when changes in its status occurred. This means that a Product Specification, defined at a given date, may reference an item in the Register at a specific point in time.

2-4.2.1 Registers within the IHO Geospatial Information (GI) Registry

The following Registers have been implemented within the IHO Geospatial Information (GI) Registry structure. A full description of these Registers is included in the accompanying references:

- Concept Register (S-100 Part 2a);
- Data Dictionary Register (S-100 Part 2a);
- Portrayal Register (S-100 Part 2b);
- Metadata Register (not currently described - to be included in a future Edition of S-100);
- Product Specification Register (not currently described - to be included in a future Edition of S-100); and
- Producer Code Register (not currently described - to be included in a future Edition of S-100).

A description of the operational and management procedures for the IHO GI Registry, expanding further on the general concepts described in the following clauses, is included in IHO Publication S-99 – *Operational Procedures for the Organization and Management of the IHO Geospatial Information (GI) Registry*.

2-5 Roles and responsibilities in the management of Registers

2-5.1 Register Owner

The Register Owner is an organization that:

1) establishes one or more Registers within a Registry;
2) has primary responsibility for the management, dissemination, and intellectual content of those Registers;
3) may appoint another organization to serve as the Register Manager; and
4) shall establish a procedure to process proposals and appeals made by Submitting Organizations.

2-5.2 Register Manager

The Register Manager is responsible for the administration of a Register. This includes:

1) coordinating with other Register Managers, Submitting Organizations, the related Control Body, Register Owner and the Registry Manager;
2) maintaining items within the Register;
3) maintain and publish a List of Submitting Organizations;
4) distributing an information package containing a description of the Register and how to submit proposals; and
5) providing periodic reports to the Register Owner and/or the Control Body. Each report shall describe the proposals received and the decisions taken since the last report. The interval between those reports must not exceed 12 months.

A Register Manager may manage multiple Registers.

2-5.3 Register User

A Register User is any person or organization interested in accessing or determining the content of a Register.

2-5.4 Domain Control Body

The Domain Control Body (DCB) for a Register is a group of technical experts appointed by a Register Owner to decide on the acceptability of proposals for changes to the content of a Register. The group must comprise experts in the related field that makes up the contents of the Register. As such, the DCB consists of at least one representative from each of the Domains contributing to the content of the Register.

2-5.5 Executive Control Body

The Executive Control Body (ECB) must consist of a representative of each of the Domains within the Register. The ECB will monitor and advise the Register Manager(s) and act as arbiters for any decisions or disputes in the Register process. In the event that a resolution cannot be achieved, the ECB may request a decision from the IHO Hydrographic Services and Standards Committee (HSSC).

2-5.6 Submitting Organizations

2-5.6.1 Eligible Submitting Organizations

A Submitting Organization is an organization that is qualified under criteria determined by the Register Owner to propose changes to the content of a Register. The Register Manager must determine whether a Submitting Organization is qualified in accordance with the criteria established by the Register Owner. An approved Submitting Organization shall have at least one representative appointed to submit proposals to the Register on its behalf (multiple representatives of a Submitting Organization may be appointed in consultation with the Register Manager).

2-5.7 Processing of Proposals

Submitting Organizations may submit requests for addition, clarification, supersession, and retirement of registered items.

2-5.7.1 Addition of registered items

Addition is the insertion into a Register of a new item that describes a concept not adequately described by a valid or processing item already in the Register.

2-5.7.2 Clarification of registered items

Clarifications correct errors in spelling, punctuation, grammar or improvements to content or wording. A clarification shall not cause any substantive semantic change to a registered item. The three characteristics that may be clarified are definition, other references, and remarks.

2-5.7.3 Supersession of registered items

Supersession of an item means any proposal that would result in a substantive semantic change to an existing valid item, such as a change to the name of an item or its camelCase identifier; or a change in the portrayal of an item in the Portrayal Register. Supersession shall be accomplished by including one or more new items in the Register with new identifiers and a more recent date. The original item shall remain in the Register but must include the date at which it was superseded, and a reference to the items that superseded it.

2-5.7.4 Retirement of registered items

Retirement shall be affected by leaving the item in the Register; marking it retired; and including the date of retirement.

2-6 The Register Manager shall

1) Receive proposals from Submitting Organization representatives;
2) Review proposals for completeness;
3) Return proposals to the Submitting Organization representative if incomplete;
4) Check within the Register for similar proposals; and if similar, the Register Manager shall contact the Submitting Organization representative;
5) Coordinate proposals with other Register Managers within two calendar weeks from the date received;
6) Generate a proposal management record, with the status set to 'transferred'; and
7) Initiate the approval process.

2-6.1 Proposal legitimacy

The Register Manager shall use the following criteria to determine if the proposal is complete and reject the proposal if:

1) The submitter is not a qualified Submitting Organization representative;
2) The proposed item does not belong to an item class assigned to this Register Manager;
3) The proposed item does not fall within the scope of the Register; or
4) The proposed item has already been proposed.

2-7 Proposal process

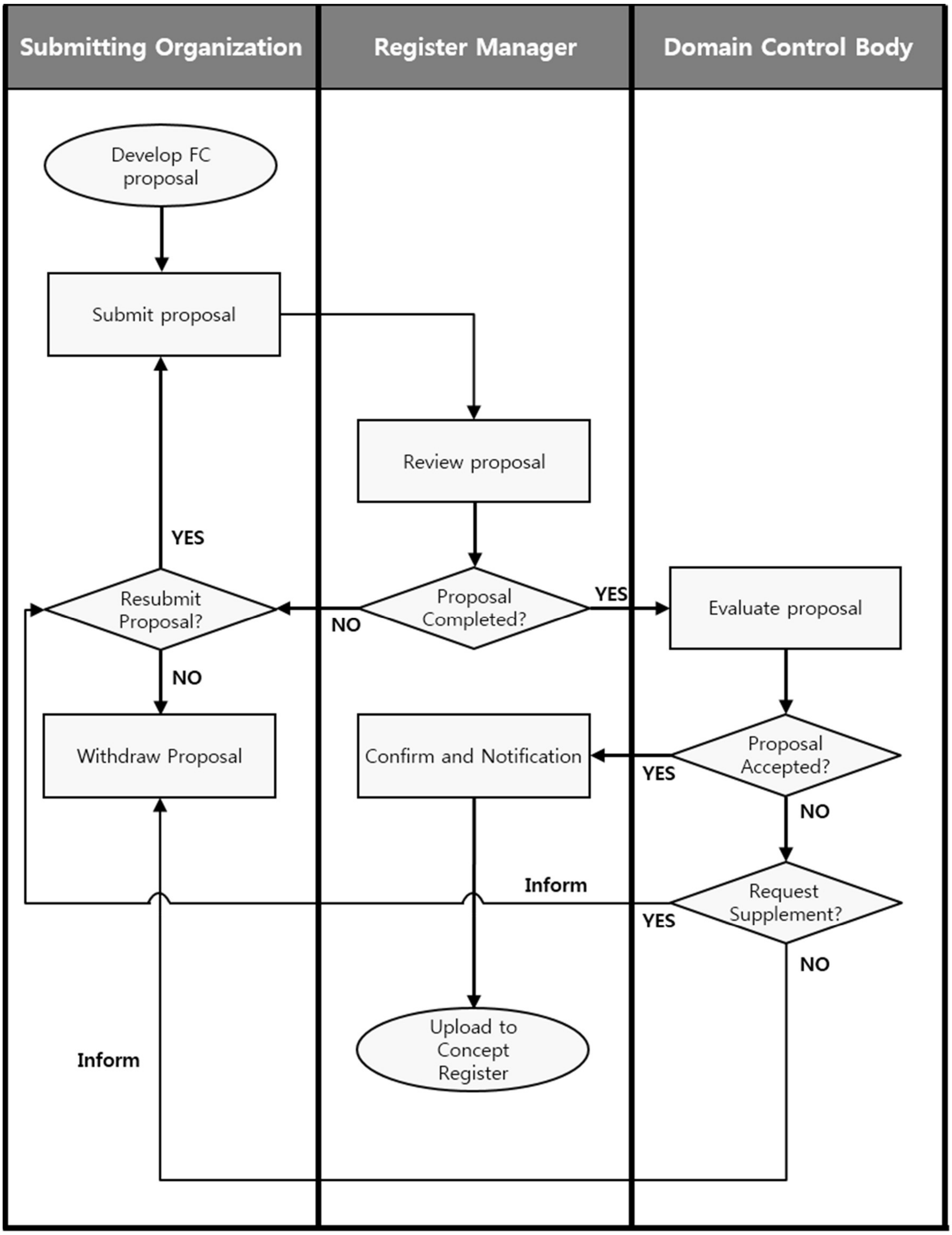


Figure 2-1 – Proposal process – Concept Register

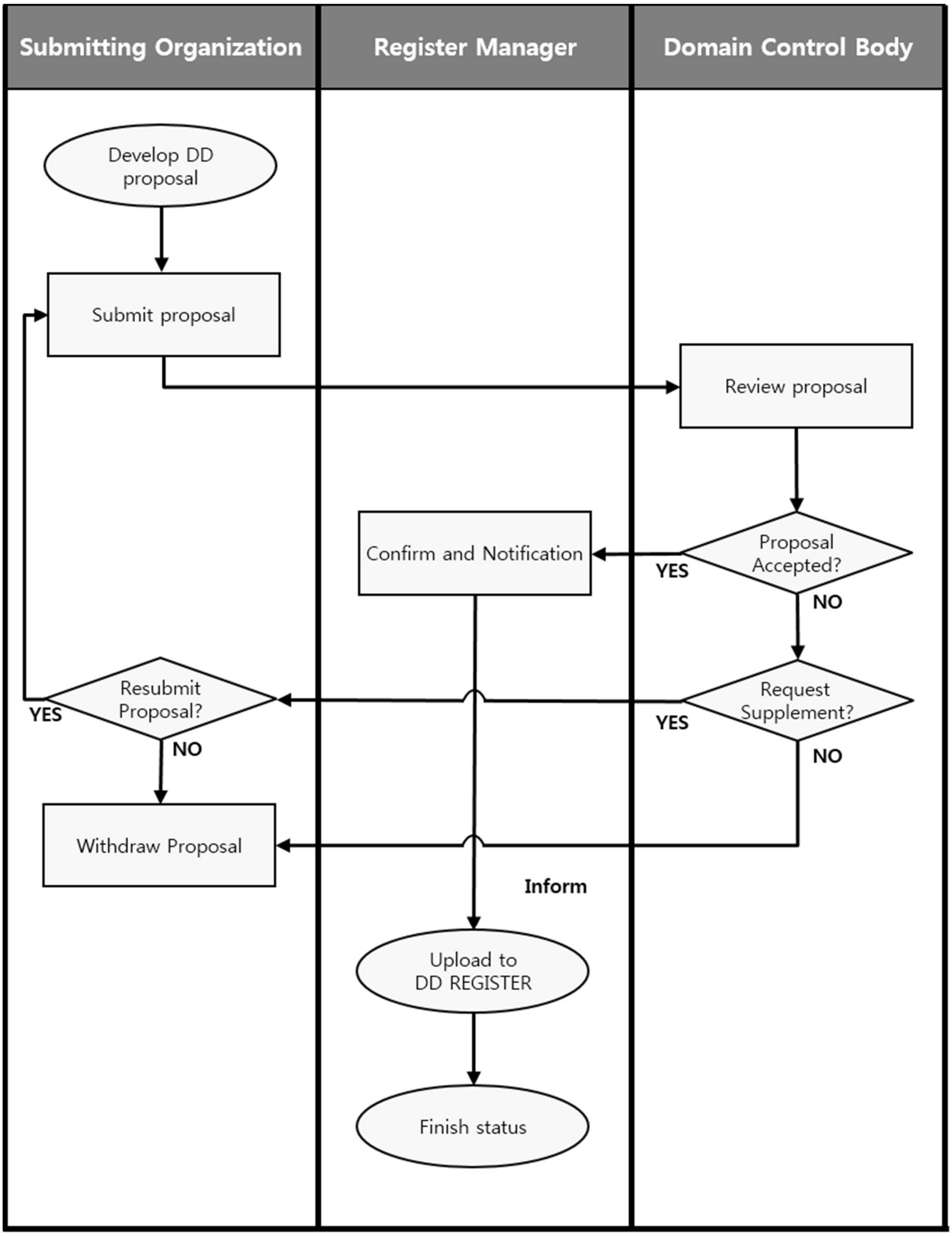


Figure 2-2 – Proposal process – Data Dictionary Register

2-7.1 Approval process

The process for determining the acceptability of proposals is illustrated in Figures 2-1 and 2-2 above. It must be completed within a time period specified by the Register Owner, in accordance with S-99.

The Register Manager shall ensure the following:

1) If the proposal is for clarification or retirement of a Register item, set the *itemStatus* of the item to ‘processing'; and forward the proposal to the Register Domain Control Body; or

2) If the proposal is for registration of a new item or supersession of an existing Register item:
    a) Assign an *itemIdentifier* to the new or superseding item;
    b) Set the *itemStatus* of the item to 'processing', and
    c) forward the proposal to the Register Domain Control Body.

The Register Domain Control Body shall:

1) Decide to accept the proposal without change; to accept the proposal subject to changes negotiated with the Submitting Organization; or not to accept the proposal. Criteria for not accepting a proposal include:
    a) The specification of the item is incomplete or incomprehensible;
    b) An identical or very similar item already exists in the Register or in another Register of the Registry;
    c) The proposed item does not belong to an item class included in the Register;
    d) The proposed item does not fall within the scope of this Register; or
    e) The justification for the proposal is inadequate.

2) Inform the Register Manager of the decision, and the rationale for the decision, within a time limit specified by the Register Owner, in accordance with S-99.

The Register Manager shall:

1) Serve as the point of contact if there is a need for negotiations between the Submitting Organization and the Register Domain Control Body regarding changes to the proposal that are specified by the Domain Control Body as a condition of acceptance; and

2) Inform the Submitting Organization representative of the results of processing the proposal.

If the decision of the Register Domain Control Body is positive, the Register Manager shall, in accordance with policies for the Register:

1) Complete the proposal management record with *proposalStatus* set to 'accepted'; and *dateAmended* to the date of the Domain Control Body's decision;

2) Make approved changes to the content of the Register item; and

3) Set the Register item *itemStatus* to 'valid'. For supersession or retirement proposals, the replaced or retired item has *itemStatus* set to 'superseded' or 'retired', as appropriate.

If the decision of the Register Domain Control Body is negative, the Register Manager shall:

1) Update the proposal management record by setting *proposalStatus* to 'rejected'; and *dateAmended* to the date of the Domain Control Body's decision; and

2) Inform the Submitting Organization of the deadline for appealing the decision of the Domain Control Body.

Submitting Organization representatives shall:

1) Negotiate with the Register Domain Control Body through the Register Manager, with regard to changes to their proposal that are specified by the Domain Control Body as a condition of acceptance; and

2) Make known within their respective Submitting Organization communities or organizations the decisions taken on proposals by the Domain Control Body as transmitted to them by the Register Manager.

The Register Manager shall:

1) Disseminate the results of the approval process to the public.

2-7.1.1 Withdrawal of proposals

Submitting Organization representatives may decide to withdraw a proposal at any time during the approval process.

The Register Manager shall:

1) Change the proposal management record *proposalStatus* from 'not yet determined' to 'withdrawn'.

2-7.1.2 Appeals

A Submitting Organization representative may appeal to the Register Executive Control Body if it disagrees with the decision of a Register Domain Control Body to reject a proposal for addition, clarification, supersession or retirement of an item in the Register. An appeal shall contain at a minimum a description of the situation; a justification for the appeal; and a statement of the impact if the appeal is not successful. The appeal process is included in the overall proposal process as shown in Figures 2-1 and 2-2 above.

The Submitting Organization shall:

1) Determine if the decision regarding a proposal for registration is acceptable; and
2) If not, submit an appeal to the Register Manager.

The Register Manager shall:

1) Forward the appeal to the Register Executive Control Body.

The Register Executive Control Body shall:

1) Process the appeal in conformance with its established procedures;
2) Decide whether to accept or reject the appeal; and
3) Return the result to the Register Manager.

The Register Manager shall:

1) Update the proposal management record fields *proposalStatus* and *dateAmended*;
2) Update the Register item *itemStatus*; and
3) Provide the results of the decision to the Register Domain Control Body and to the Submitting Organization representative.

The Submitting Organization representative shall:

1) Make the results of the appeal known within their Submitting Organization community or organization.

2-7.2 List of Submitting Organizations

The Register Manager shall maintain and publish a Register-specific list of all qualified Submitting Organizations that may submit proposals for changes to the content of each Register that it manages. Each list shall include the name and contact information of the representative(s) for each Submitting Organization. The Registry shall contain an application to become a Submitting Organization. The Register Owner will be responsible for accepting or rejecting the application.

2-7.3 Publication

The Registry Manager shall ensure that information about valid, clarified, superseded, or retired items in the Register is readily available to users. The method for providing this information may depend upon the requirements of the members of the user community.

2-7.4 Integrity

The Register Manager shall ensure that, for each Register being managed:
1) All aspects of the registration process are handled in accordance with good business practice;
2) The content of the Register is accurate; and
3) Only authorised persons can make changes to the Register content.

The Registry Manager shall ensure the security and integrity of the Registry using IT best practices.

2-8 Register Schema

2-8.1 Introduction

The schema specified in this clause describes the structure of an IHO Geospatial Information Register.

Information about the Register and items in the Register shall be:
1) Accessible through an on-line interface to the Register;
2) Included in any copy of the Register; and
3) Included in any information package about the Register.

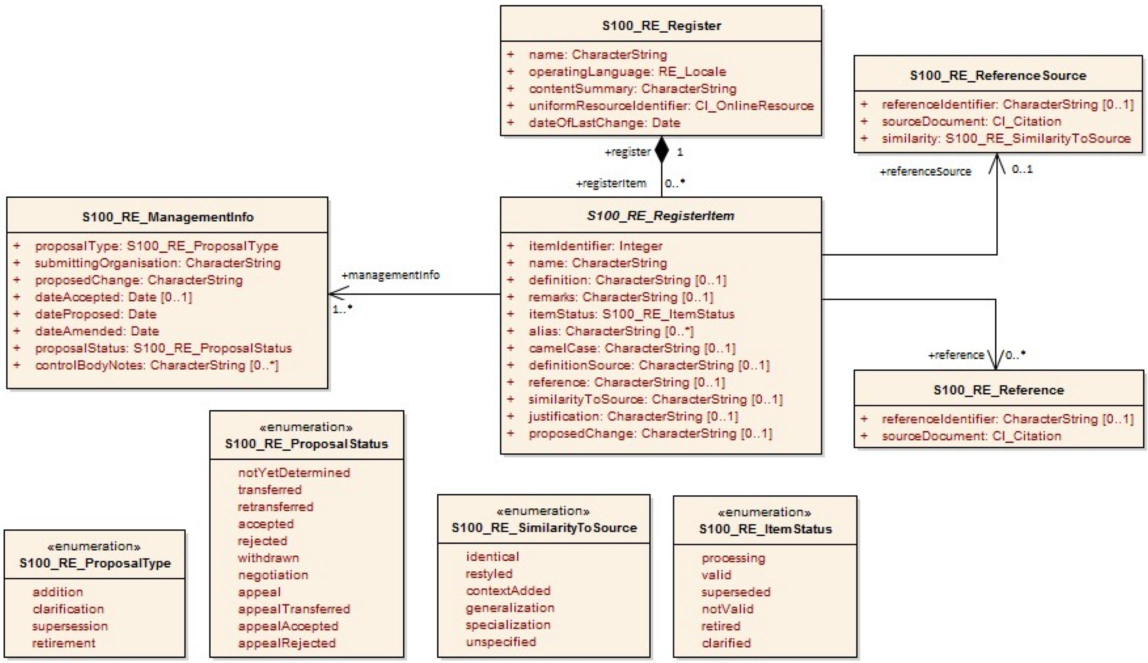


Figure 2-3 – The Register schema

Page intentionally left blank

2-8.2 S100_RE_Register

The class S100_RE_Register specifies information about the Register itself.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_RE_Register		-	-	
Attribute	name	The name of the Register	1	CharacterString	Unique within the Registry
Attribute	operatingLanguage	The language used in this Register	1	RE_Locale	
Attribute	contentSummary	Summary of the content	1	CharacterString	
Attribute	uniformResourceIdentifier	The link to the interface of the Register in the Internet	1	CI_OnlineResource	
Attribute	dateOfLastChange	The date when the last change was made to this Register	1	Date	
Association	registerItem	The items of the Register	0..*	S100_RE_RegisterItem	

2-8.3 S100_RE_RegisterItem

The class S100_RE_RegisterItem carries the characteristics that are common to all types of registered items. Domain specific extensions may be added in the appropriate part of S-100; for example, Part 2a – Concept and Data Dictionary Registers.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_RE _RegisterItem		-	-	Class is abstract
Attribute	itemIdentifier	Each item has its own unique identifier in a Register	1	Integer	
Attribute	name	Succinct expression of the item concept it denotes	1	CharacterString	
Attribute	definition	Shall be a precise statement of the nature, properties, scope, or essential qualities of the concept as realized by the item	0..1	CharacterString	Mandatory for the Concept, Data Dictionary, Portrayal and Metadata Registers
Attribute	remarks	Supplementary information	0..1	CharacterString	
Attribute	itemStatus	The state in which a registered item exists	1	S100_RE_ItemStatus	
Attribute	alias	Equivalent name(s) used for the item	0..*	CharacterString	
Attribute	camelCase	Identifier of the item using camelCase notation	0..1	CharacterString	

Role Name	Name	Description	Mult	Data Type	Remarks
Attribute	definitionSource	The source of the definition	0..1	CharacterString	
Attribute	reference	An identifier of the place in the definitionSource from which the definition is derived	0..1	CharacterString	
Attribute	similarityToSource	The type of change that has been made to a definition relative to the definition as it exists in the definition source	0..1	CharacterString	
Attribute	justification	Primary reason for the proposal including how it is proposed to be used	0..1	CharacterString	Inherited from S-100_RE_ManagementInfo
Attribute	proposedChange	The text of the proposed change	0..1	CharacterString	Inherited from S-100_RE_ManagementInfo
Association	register	The Register that contains the item	1	S100_RE_Register	
Association	referenceSource	The source information the item definition was taken from.	0..1	S100_RE_ReferenceSource	
Association	reference	Reference to other relevant standards or documents	0..*	S100_RE_Reference	For example INT1 or M4
Association	managementInfo	Sets of information describing the management of the item in the Register	1..*	S100_RE_ManagementInfo	

The camelCase must:

1) Be compound words in which the words are joined without spaces and are capitalized within the compound.
2) Be unique within the Registry.
3) Conform to UTF-8 character encoding (refer ISO/IEC 10646) with uppercase characters A-Z, lowercase characters a-z and numbers 0-9.
4) Concepts must begin with lowercase a-z.

Example 1 beaconCardinal is the Camel Case identifier for the concept Beacon Cardinal

Example 2 categoryOfLandmark is the Camel Case identifier for the concept Category of Landmark

2-8.4 S100_RE_ItemStatus

The enumeration S100_RE_ItemStatus identifies the registration status of a Register item.

Item	Name	Description	Remarks
Enumeration	S100_RE_ItemStaus		

Literal	processing	The item has been entered into the Register, but the Domain Control Body is evaluating the proposal	
Literal	valid	The item has been accepted, is recommended for use, and has not been superseded or retired	
Literal	superseded	The item has been superseded by one or more items and is no longer recommended for use	
Literal	notValid	The item has been entered into the Register, but the Domain Control Body has not accepted the proposal to add it	
Literal	retired	A decision has been made that the item is no longer recommended for use. It has not been superseded by another item	
Literal	clarified	The item has been clarified and is no longer recommended for use	

2-8.5 S100_RE_ReferenceSource

The class S100_RE_ReferenceSource specifies information about the source of a Register item taken from an external document or Register.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_RE_ReferenceSource		-	-	
Attribute	referenceIdentifier	An identifier of the place in the source document that is referenced	0..1	CharacterString	
Attribute	sourceDocument	The source document	1	CI_Citation	
Attribute	similarity	Indicates how the definition is related to the source document	1	S100_RE_SimilarityToSource	

2-8.6 S100_RE_SimilarityToSource

The enumeration S100_RE_SimilarityToSource identifies the type of change that has been made to an item specification relative to an item specification in an external source.

Item	Name	Description	Remarks
Enumeration	S100_RE_SimilarityToSource		

Item	Name	Description	Remarks
Literal	identical	No change has been made to the definition	
Literal	restyled	The style of the definition has been changed to match the style and structure of other definitions in the Register that has imported the definition	
Literal	contextAdded	The definition includes information about its context that is not explicit in the specification in the external source	
Literal	generalization	The definition of the register item has been generalized to have a broader meaning than the item specified in the external source	
Literal	specialization	The definition of the Register item has been specialized to have a narrower meaning than the item specified in the external source	
Literal	unspecified	The nature of the differences between the Register item and the similar item in the external source is unspecified	

2-8.7 S100_RE_Reference

The class S100_RE_Reference specifies information about the source and/or lineage of a specific Register item derived from an external document or Register.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_RE_Reference		-	-	
Attribute	referenceIdentifier	An identifier of the place in the source document that is referenced	0..1	CharacterString	
Attribute	sourceDocument	The source document	1	CI_Citation	

2-8.8 S100_RE_ManagementInfo

The class S100_RE_ManagementInfo specifies the management record of a Register item.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_RE_ManagementInfo		-	-	
Attribute	proposalType	The type of the proposal	1	S100_RE_ProposalType	
Attribute	submittingOrganisation	The proposal's sponsor	1	CharacterString	
Attribute	proposedChange	The text describing the proposed change	1	CharacterString	

Role Name	Name	Description	Mult	Data Type	Remarks
Attribute	justification	Primary reason for the proposal including how it is proposed to be used	1	CharacterString	
Attribute	dateAccepted	Date the proposal was accepted	0..1	Date	
Attribute	dateProposed	Date the proposal was made	1	Date	
Attribute	dateAmended	Date the proposal was adjudicated	1	Date	
Attribute	proposalStatus	Provides values for describing the disposition of a proposal to add or modify a Register item	1	RE_ProposalStatus	
Attribute	controlBodyNotes	Supplementary management information	0..*	CharacterString	

2-8.9 S100_RE_ProposalType

The enumeration S100_RE_ProposalType species the type of proposal for a Register item.

Item	Name	Description	Remarks
Enumeration	S100_RE_ProposalType		
Literal	addition	The item is to be added to the Register	
Literal	clarification	A non-substantive change to an item in the Register	
Literal	supersession	The item has been superseded by another item and is no longer recommended for use.	
Literal	retirement	A decision has been made that the item is no longer recommended for use. It has not been superseded by another item	

2-8.10 S100_RE_ProposalStatus

The enumeration S100_RE_ProposalStatus specifies the disposition of a proposal to add or change a Register item.

Item	Name	Description	Remarks
Enumeration	S100_RE_ProposalStatus		
Literal	notYetDetermined	The submitting Organization has submitted the proposal	
Literal	transferred	The Register Manager has requested the Domain Control Body to review the proposal	

Item	Name	Description	Remarks
Literal	retransferred	The Register Manager has requested the Submitting Organization to supplement the proposal	
Literal	accepted	The Control Body has accepted the proposal	
Literal	rejected	The Control Body has not accepted the proposal	
Literal	withdrawn	The Submitting Organization has withdrawn the proposal	
Literal	negotiation	The Domain Control Body has requested the Submitting Organization to supplement the proposal	
Literal	appeal	The Submitting Organization has initiated an appeal	
Literal	appealTransferred	The Register Manager has transferred the appeal to the Executive Control Body	
Literal	appealAccepted	The Executive Control Body has accepted the appeal	
Literal	appealRejected	The Executive Control Body has not accepted the appeal	

S-100 – Part 2a

Concept and Data Dictionary Registers

Page intentionally left blank

Contents

Page intentionally left blank

2a-1 Scope

The IHO Geospatial Information (GI) Registry contains a number of Registers, some of which are based on the contents of the Concept Register. The Concept Register specifies hydrographic core conceptual information (definitions, camelCase, etc) that may be used to describe geographic or meta data information. The use of a Register to store hydrographic definitions significantly improves the IHOs ability to manage and extend multiple products based on S-100 which can be made available for use in a relatively short timescale. As such, the Register supports wider use of registered items by making them publicly available; and increases their visibility to potential users. The Concept Register is the primary resource where all registered concepts are stored and managed as “stateless” concepts (that is, items are not assigned a type and there is no defined binding of concepts to other concepts within the Register). Each concept shall be included as a single instance in the Register and will be used as the common source from which Data Dictionary Register and Meta Data Register concepts are derived and used to model features, attributes etc. for use in S-100 based Product Specifications.

The Data Dictionary Register expands on the concepts stored in the Concept Register by including the assignment of item types and feature binding in discrete Domains within the Register. This allows S-100 based Product Specification developers to develop their data models to best suit their specific requirements for representation of the real world.

This Part describes the content of the Concept and Data Dictionary Registers and specifies procedures to be followed in establishing, maintaining, and publishing Registers of unique, unambiguous and permanent identifiers that are assigned to items of geographic, hydrographic and metadata information. In order to accomplish this purpose, this Part specifies elements of information that are necessary to provide identification and definitions to the registered items.

2a-1.1 Conformance

This profile conforms to conformance class 2 of ISO 19106:2004. The following is a brief description of the specializations and generalizations where the profile differs from ISO 19126:2008.

1) A new class, S100_CD_Information is introduced.
2) New classes, S100_CD_Feature and S100_CD_Information are introduced.
3) A new class, S100_CD_AttributeConstraints is introduced.
4) The class FC_FeatureAttribute is specialized to be the abstract class *S100_CD_Attribute*.
5) New classes, S100_CD_SimpleAttribute and S100_CD_ComplexAttribute are introduced.
6) The classes CD_InheritanceRelation, CD_FeatureOperation CD_Binding, CD_Constraint and CD_BoundFeatureAttribute are not used.

2a-2 Normative references

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

ISO 19135:2005, *Geographic Information – Procedures for registration of items of geographic information*

ISO 19126:2009, *Geographic Information – Feature concept dictionaries and registers*

ISO 8601:2004, *Data elements and interchange formats - Information interchange - Representation of dates and times*

ISO/IEC 10646:2017, *Information Technology – Universal Coded Character Set (UCS)*

RFC 3986, *Uniform Resource Identifier (URI): Generic Syntax*. T. Berners-Lee, R. Fielding, L. Masinter. Internet Standard 66, IETF. URL: http://www.ietf.org/rfc/rfc3986.txt or http://www.rfc-editor.org/info/std66

RFC 2141, *URN Syntax*. R. Moats. IETF RFC 2141, May 1997. URL: https://www.ietf.org/rfc/rfc2141.txt

W3C *XML Schema Part 2: Datatypes* Second Edition, Appendix F *Regular Expressions*. URL: https://www.w3.org/TR/xmlschema-2/#regexs

2a-3 General concepts

2a-3.1 Register

As described in Part 2, a Register is simply a managed list. It is easier to maintain than a fixed document, because new items can be added as needed to the Register, and existing items in the Register can be clarified, superseded or retired. Each Register item has one or more dates associated with it that indicate when changes in its status occurred. This means that a Product Specification, defined at a given date, may reference an item in the Register at that specific point in time.

2a-3.2 Relationship to the ISO Register and Feature Concept Dictionary Models

The realization of ISO 19126 and ISO 19135 types in the IHO GI Registry is shown in Figure 2a-1 below. This Figure depicts the ISO types (elements with grey backgrounds); the implemented S-100 Registry model classes (elements with tan backgrounds); and two of the Registers comprising the IHO GI Registry (elements with white backgrounds). The Registers themselves are implemented within different sections of the IHO GI Registry web site, each accessible through a site navigation menu.

The ISO type “Feature Concept Dictionary”, which contains definitions of feature and attribute types, is implemented as the “Data Dictionary Register” in the IHO GI Registry (S100_DataDictionaryRegister in Figure 2a-1). The Data Dictionary Register contains definitions of features, information types, attributes (both simple and complex), and enumerated values for enumeration and S100_Codelist attributes (see S-100 Part 1, clauses 1-4.7 and 1-4.8). The Data Dictionary Register is further described in clauses 2a-3.4 and 2a-4.1.2.

The IHO GI Registry also implements a “Concept Register” (S100_ConceptRegister in Figure 2a-1), which can be understood as a kind of glossary of terms with definitions and sources. Individual entries in the Concept Register are represented by the class S100_Concept. The Concept Register is separate from the Data Dictionary Register. However, entries in the Data Dictionary Register are derived from entries in the Concept Register. The Concept Register is further described in clauses 2a-3.3 and 2a-4.1.1.

Registers for associations and roles are not currently implemented within the IHO GI Registry and the corresponding ISO types are therefore not included in Figure 2a-1. Also, the IHO GI Registry includes additional Registers (for Portrayal, Producer Codes, Product Specifications, and Metadata) which are not shown in Figure 2a-1.

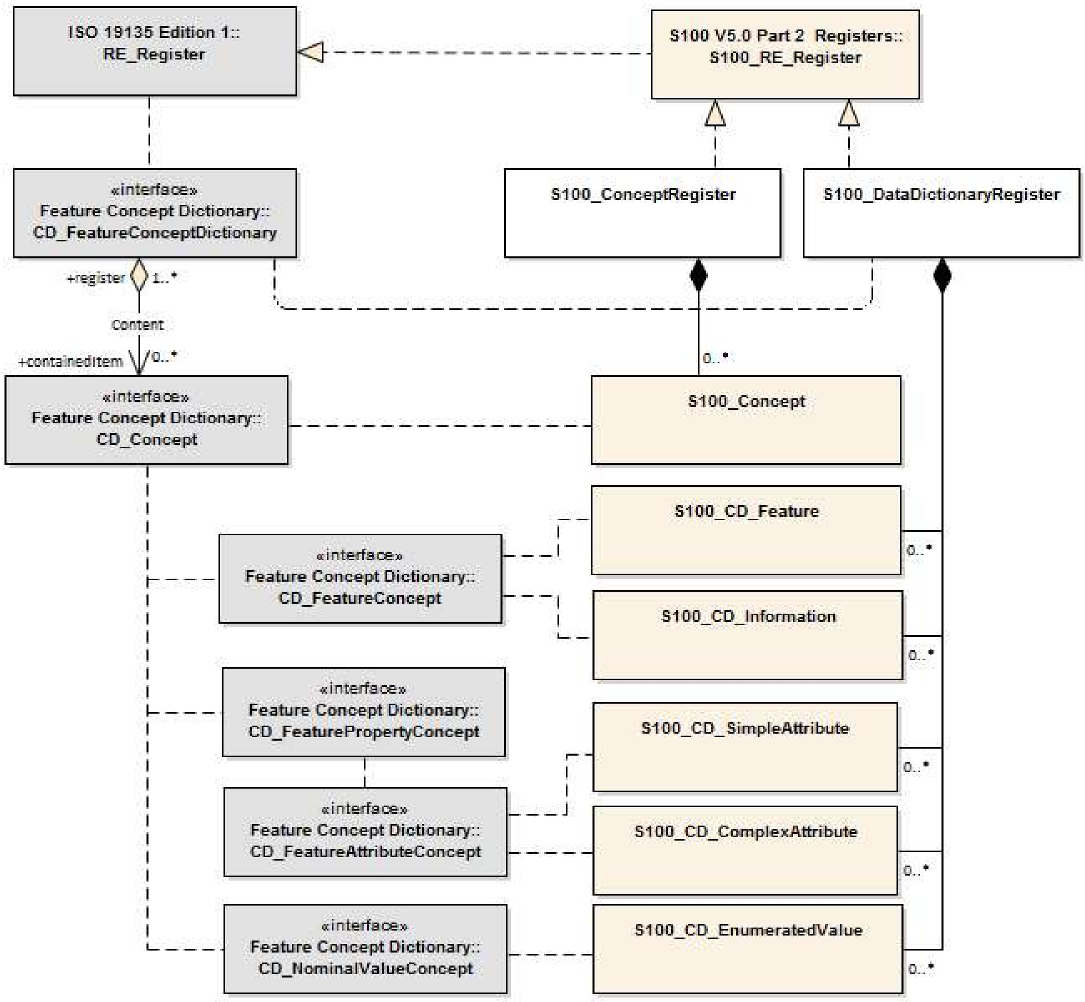


Figure 2a-1 – Relationships of the S-100 Registry Classes to ISO 19126/ISO 19135 Model

2a-3.3 Concept Register

A Concept Register specifies unique, independent sets of definitions of concepts that may be used to describe geographic, hydrographic, and metadata information. These concepts as registered in a Concept Register may then be used within a Data Dictionary or Meta Data Register to develop a Feature Catalogue. Unlike a Data Dictionary Register, a Concept Register does not make associations; or define type or bindings of concepts to other concepts. From this perspective, registered items within the Concept Register are essentially “stateless”, which allows for flexibility of the use of the concepts for suitable data modelling to satisfy the requirements of Product Specifications.

Such Registers of geospatial information may serve as sources of reference for similar Registers established by other geographic information communities as part of a system of cross-referencing.

2a-3.4 Data Dictionary Register

A Data Dictionary Register specifies independent sets of definitions of features, attributes, enumerated values, and information types that may be used to describe geographic, hydrographic, and metadata information. A Data Dictionary Register may be used to assign an item defined in a Concept Register a type (for example feature, attribute and enumerate value); and define recommended associations and attribute/feature bindings to facilitate the development of Feature Catalogues. Items in a Concept Register can only be registered once against each type in a Data Dictionary Register in order to support interoperability.

Such Data Dictionaries of geospatial information may serve as sources of reference for similar Data Dictionaries or Registers established by other geographic information communities as part of a system of cross-referencing.

2a-3.5 Feature Catalogue

A Feature Catalogue is a document that describes the content of a data product. It uses item types, for example features and attributes, from one or more Data Dictionaries and binds them together. In addition constraints, units of measurement and format description of attributes can be specified. Feature Catalogues are described in detail in S-100 Part 5.

2a-4 IHO Concept and Data Dictionary Registers

2a-4.1 Detail of registered items

2a-4.1.1 IHO Concept Register

The following are the details available in a Concept Register to describe and manage hydrographic, marine-related and meta data information:

1) Item Name
2) Item Identifier
3) Item Status
4) Alias
5) Camel Case
6) Definition
7) Definition Source
8) Definition Reference
9) Similarity to Source
10) Remarks
11) Proposal Type
12) Successor
13) Predecessor
14) Submitting Organization
15) Date Proposed
16) Date Accepted
17) Date Amended
18) Proposed Change
19) Justification

These details are derived from the classes S100_RE_RegisterItem and S100_RE_ManagementInfo (see clause 2a-4.2 below and Part 2, clauses 2-8.3 and 2-8.8).

2a-4.1.2 IHO Data Dictionary Register

The following are types of items, as derived from a Concept Register, which may be registered within a Data Dictionary Register:

1) Feature – abstraction of real world phenomena.
2) Attribute – characteristic of a feature concept.

3) Enumerated Value – one of a set of mutually exclusive values constituting the domain of an attribute.
4) Information – an identifiable object that contains attributes, associations to other information concepts, but no spatial information.
5) Codelist – an open enumeration, or the identifier of a vocabulary (mapping between codes, labels and definitions).

2a-4.2 Data model of the IHO Concept and Data Dictionary Registers

2a-4.2.1 UML Model

The following Figure shows the UML information model of the IHO Concept and Data Dictionary Registers:

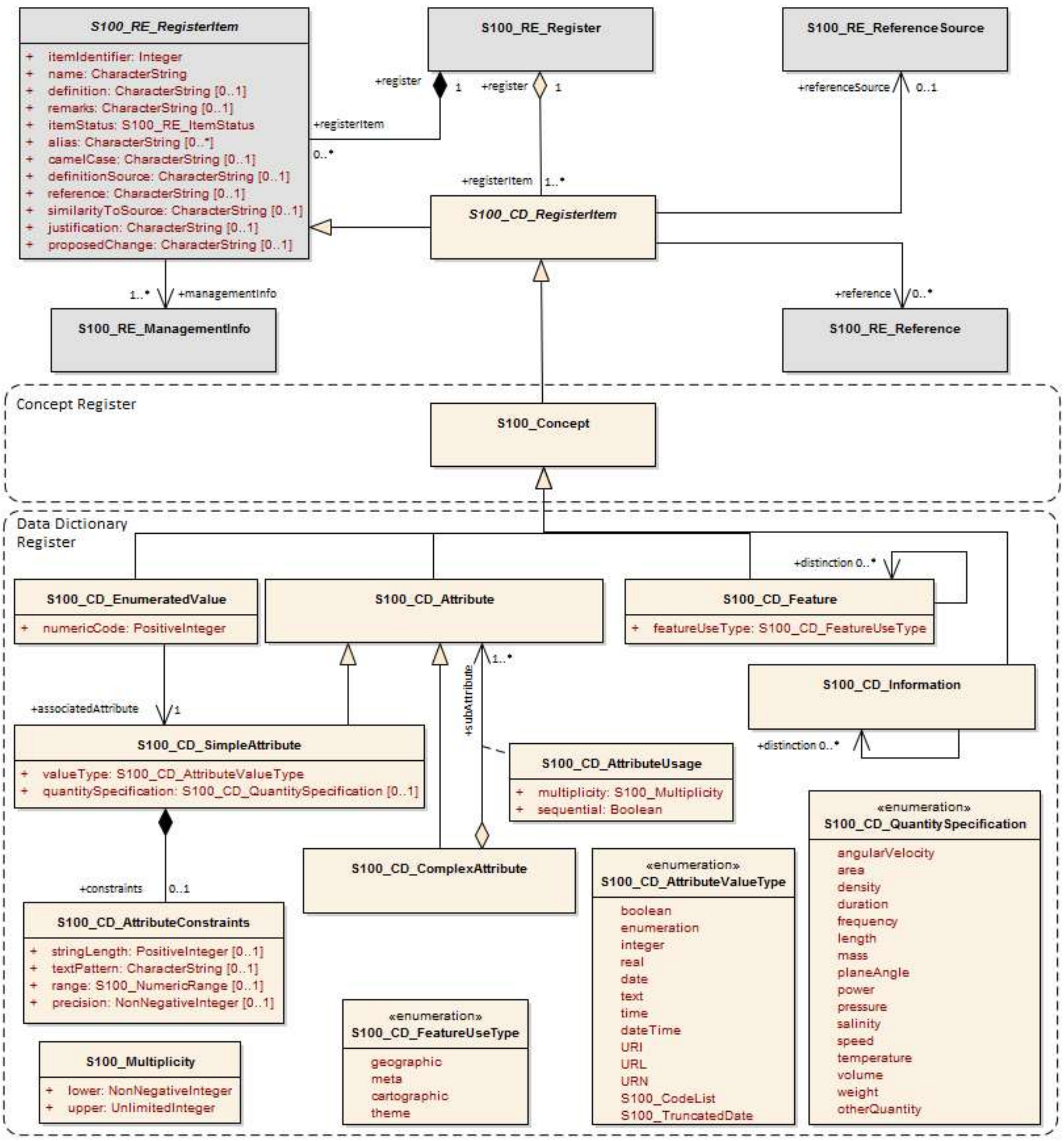


Figure 2a-2 – IHO Concept and Data Dictionary Registers

Page intentionally left blank

2a-4.2.2 S100_RE_Register

The class S100_RE_Register models a Register in a Registry. Further details can be found in S-100 Part 2, clause 2-8.2.

2a-4.2.3 S100_CD_RegisterItem

The class S100_CD_RegisterItem is a specialization of the class S100_RE_RegisterItem and carries the characteristics that are common to all types of registered items listed in clause 2a-4.1.1. This class is included in the model for consistency with Part 2b, in which a similar extension (S100_PR_RegisterItem) defines additional attributes.

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_CD_RegisterItem	Extension of S100_RE_RegisterItem	-	-	Abstract class

2a-4.2.4 S100_CD_Feature

This class is derived from S100_RE_RegisterItem via intermediate super-classes. It defines the following additional properties:

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_CD_Feature	A feature type in a Data Dictionary	-	-	Derived from S100_Concept
Attribute	featureUseType	The intended use of a feature type	1	S100_CD_FeatureUseType	
Association role	distinction	References to feature types that this feature type is distinct from	0..*	S100_CD_Feature	
Association	conceptReference	References S100_Concept as the base class	1	S100_Concept	

NOTE The attribute camelCase is mandatory for this class.

2a-4.2.5 S100_CD_FeatureUseType

Item	Name	Description	Remarks
Enumeration	S100_CD_FeatureUseType	Categories of feature types	
Literal	geographic	carries the descriptive characteristics of a real world entity	
Literal	meta	Delineates geographic location where meta information is applicable" distinct from an Information Type which carries information related to features which are related	
Literal	cartographic	carries information about the cartographic representation (including text) of a real world entity	
Literal	theme	Grouping features thematically	

2a-4.2.6 S100_CD_Attribute

Attributes may either be simple or complex. A simple attribute carries a specific value such as a date. A complex attribute is an aggregation of other attributes either simple or complex. Examples of complex attributes are in Appendix 2a-A. This class is derived from S100_RE_RegisterItem via intermediate super-classes and describes the common characteristics of all attribute types.

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_CD_Attribute	Base class of all attribute types in a Data Dictionary	-	-	Derived from S100_Concept
Association	conceptReference	References S100_Concept as the base class	1	S100_Concept	

NOTE The attribute camelCase is mandatory for this class.

2a-4.2.7 S100_CD_SimpleAttribute

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_CD_SimpleAttribute	A simple attribute type in a Data Dictionary	-	-	Derived from S100_CD_Attribute
Attribute	valueType	Describes representation, interpretation and structure of values	1	S100_CD_AttributeValueType	See below
Attribute	quantitySpecification	Specification of the quantity, for example length, volume, depth, weight etc	0..1	S100_CD_QuantitySpecification	
Association	constraints	Constraints of the attribute type	0..1	S100_CD_AttributeConstraints	Must be consistent with dataType

If the *valueType* is S100_Codelist exactly one of the following must be true:

1) There is an associated S100_RE_Reference with the namespace of a dictionary that is listed in the IHO GI Registry.
2) There is at least one S100_CD_EnumeratedValue associated to the attribute.

Condition 1 identifies the dictionary for codelists of type “open dictionary” or “closed dictionary”. Condition 2 provides the enumerated value(s) for codelists of type “open enumeration”. The precise codelist type is determined in individual Product Specifications.

2a-4.2.8 S100_CD_QuantitySpecification

Item	Name	Description	Remarks
Enumeration	S100_CD_QuantitySpecification	Types of quantity measures	Adapted from ISO 19103 Measure Types
Literal	angularVelocity	The instantaneous rate of change of angular displacement with time	From ISO 19103

Item	Name	Description	Remarks
Literal	area	The measure of the physical extent of any two-dimensional geometric object	From ISO 19103
Literal	density	Mass per unit volume; number per unit area. Also: specific gravity (S-32). Density of soundings is the intervals between lines of sounding and soundings in the same line (S-32)	“Density” can be used in different senses, the unit of measure and attribute definition must make it clear which is intended
Literal	duration	Interval of time	
Literal	frequency	Number of vibrations or cycles per unit time	IHO S-32
Literal	length	The longest dimension of an object; distance measured along a line or curve	
Literal	mass	A numerical measure of the inertia of an object; the quantity of matter which a body contains, irrespective of its bulk or volume	
Literal	planeAngle	The amount of rotation needed to bring one line or plane into coincidence with another, generally measured in radians or degrees	From ISO 19103 “angle”
Literal	power	Rate of doing work or transferring energy; magnification	S-32 refers “power” to “magnifying power: the ratio of the apparent length of a linear dimension as seen through an optical instrument to that seen by the unaided eye”. The unit of measure and attribute definition must make it clear which sense is intended
Literal	pressure	Force per unit area	
Literal	salinity	A measure of the quantity of dissolved salts	IHO S-32 (abbrev.)
Literal	speed	Rte of change of position with time	Usually calculated using the simple formula, the change in position during a given time interval. Speed is a scalar physical quantity, having magnitude but not direction. Contrast to “velocity” which is a vector quantity having both magnitude and direction. (Adapted from ISO 19103 “velocity”)
Literal	temperature	The intensity or degree of heat	IHO S-32
Literal	volume	The measure of the physical space of any 3-D geometric object	From ISO 19103
Literal	weight	The force experienced by an object due to gravity	
Literal	otherQuantity	A quantity different from the other literals of this enumeration	

2a-4.2.9 S100_CD_AttributeValueType

Item	Name	Description	Remarks
Enumeration	S100_CD_AttributeValueType	Value types of simple attributes	
Literal	boolean	True or False	
Literal	enumeration	List of predetermined values that can be expanded and contracted	
Literal	integer	Numeric value with defined range, units and format	
Literal	real	Floating point number	
Literal	text	A sequence of characters	
Literal	date	Character encoding shall follow the format for date as specified by ISO 8601	
Literal	time	Character encoding shall follow the format for time as specified by ISO 8601	
Literal	dateTime	Character encoding shall follow the format for date and time as specified by ISO 8601	
Literal	URI	Character encoding shall follow the format for URI as specified by RFC 3986	
Literal	URL	Character encoding shall follow the format for URL as specified by RFC 3986	
Literal	URN	Character encoding shall follow the format for URN as defined by RFC 2141	
Literal	S100_CodeList	Open enumeration or identifier of entry in a vocabulary	
Literal	S100_TruncatedDate	Truncated format for date	

2a-4.2.10 S100_CD_AttributeConstraints

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_CD_AttributeConstraints	Constraints of a simple attribute	-	-	
Attribute	stringLength	Shall be represented as a positive integer (that is, greater than zero) that specifies the maximum number of characters that may be assigned to the text attribute type. If not specified, then the text length shall be unconstrained	0..1	PositiveInteger	

Attribute	textPattern	A character string that specifies a scheme of one or more constraints on the structure of the text values that may be assigned to the attribute. This shall be achieved by using a regular expression. W3C XML Schema Part 2: Datasets Second Edition, Appendix F (Regular Expressions) shall be used to define text patterns in this standard	0..1	CharacterString	
Attribute	range	Specifies the range of allowed numeric values	0..1	S100_NumericRange	
Attribute	precision	Specifies the precision of a real number	0..1	NonNegativeInteger	

2a-4.2.11 S100_CD_ComplexAttribute

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_CD_ComplexAttribute	A complex attribute type in a Data Dictionary	-	-	Derived from S100_CD_Attribute
Association	subAttribute	References the sub attribute	1..*	S100_CD_Attribute	Characteristics defined by S100_CD_AttributeUsage

2a-4.2.12 S100_CD_AttributeUsage

This class specifies the characteristics of the association between a complex attribute type and its sub attributes.

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_CD_AttributeUsage	Characteristics of the association between a complex attribute and its sub attributes	-	-	
Attribute	multiplicity	Number of occurrences of the sub attribute	1	S100_Multiplicity	
Attribute	sequential	Boolean value that indicates if the sub attributes of a complex attribute are in a particular order	1	Boolean	It is only applicable if a sub attribute has multiplicity > 1

2a-4.2.13 S100_CD_EnumeratedValue

This class is derived from S100_RE_RegisterItem via intermediate super-classes and describes the characteristics of an enumerated value type.

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_CD_EnumeratedValue	Characteristics of an enumerated value type in a Data Dictionary	-	-	
Attribute	numericCode	A positive integer designating the unique value in the domain	1	PositiveInteger	

Association	associatedAttribute	Specifies the attribute type item for which this is a domain value	1	S100_CD_SimpleAttribute	Applies only where class S100_CD_SimpleAttribute attribute valueType = *enumeration* or *S100_CodeList*
Association	conceptReference	References S100_Concept as the base class	1	S100_Concept	

2a-4.2.14 S100_CD_Information

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_CD_Information	Characteristics of an information type in a Data Dictionary	-	-	
Association	distinction	Similar information types that this is distinct from	0..*	S100_CD_Information	
Association	conceptReference	References S100_Concept as the base class	1	S100_Concept	

NOTE The attribute camelCase is mandatory for this class.

2a-4.2.15 S100_Concept

The class S100_Concept identifies the required information for the Concept Register. Further details can be found in clause 2a-3.3.

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_Concept	A definition of object, information or phenomena of nature	-	-	

Appendix 2a – A
Example of a complex attribute
(informative)

NOTE: The following example does not reflect the modelling of lights in any S-100 based Product Specification.

A light may have several sectors. All of them share the same light characteristic and sequence. Other common attributes are the height and the name.

All attributes describing one sector in a complex attribute are structured “Light sector”.

A complex attribute for the “Rhythm of light” is also defined.

The simple attributes used in “lightSector” are:
- sectorLimit1 (type Real)
- sectorLimit2 (type Real)
- colour (type Enumeration)
- valueOfNominalRange (type Real)

Therefore the complex attribute is:

Characteristic	Value	
name	Light sector	
definition	A sector is the part of a circle between two straight lines drawn from the centre to the circumference. (Advanced Learner’s Dictionary, 2nd Edition).	
remarks	n/a	
camelCase	lightSector	
alias	LITSEC	
Sub Attributes	**Attribute Binding**	
camelCase Identifier	**multiplicity**	**sequential**
sectorLimit1	1	n/a
sectorLimit2	1	n/a
colour	1	n/a
valueOfNominalRange	0..1	n/a

Note: The multiplicity and sequence are carried in the attribute between the complex and sub-attribute.

The “Rhythm of light” consists of:
- lightCharacteristic
- signalPeriod
- signalGroup

Characteristic	Value	
name	Rhythm of light	
definition		
remarks	n/a	
camelCase	rhythmOfLight	
alias	RHYLGT	
Sub Attributes	**Attribute Binding**	
camelCase Identifer	**multiplicity**	**sequential**
lightCharacteristic	1	n/a

signalPeriod	0..1	n/a
signalGroup	0..1	n/a

A second way of describing the rhythm of light is the "signal sequence" as it is done with the S-57 SIGSEQ attribute. A signal sequence consists of intervals where the signal is either on or off (here light or eclipse)

Characteristic	**Value**	
name	Signal sequence interval	
definition	tbd.	
remarks	n/a	
camelCase	signalSequenceInterval	
alias	SGSQIN	
Sub Attributes	**Attribute Binding**	
camelCase Identifer	**multiplicity**	**sequential**
signalStatus	1	n/a
duration	1	n/a

A Signal sequence is then just an ordered list of those intervals.

Characteristic	**Value**	
name	Signal sequence	
definition	The sequence of times occupied by intervals of light and eclipse for all "light characteristics". (Adapted from S-57 Edition 3.1, Appendix A – Chapter 2, Page 2.191, November 2000).	
remarks	n/a	
camelCase	signalSequence	
alias	SIGSEQ	
Sub Attribute	**Attribute Binding**	
camelCase Identifier	**multiplicity**	**sequential**
signalSequenceInterval	1..*	True

A light object would now consist of:

Light:

- rhythmOfLight [1..*]
- lightSector [1..*]
- signalSequence [0..1]
- objectName[0..1]
- height[0..1]

This definition would be included in the Feature Catalogue, although the definition of the attributes are derived from the Concept Register.

S-100 – Part 2b

Portrayal Register

Page intentionally left blank

Contents

Page intentionally left blank

2b-1 Scope

The IHO Geospatial Information (GI) Registry contains a number of Registers, one of which will be for portrayal. The Portrayal Register registers the components used to build Portrayal Catalogues. When provided with a Feature Catalogue and compliant data, a Portrayal Catalogue built from registered components transforms features and attributes into a portrayal.

The portrayal of data is independent of the data but closely related to the data. That is, the attributes within the data set drive the portrayal process, but there may be many different portrayals for the same data.

The use of a Register to store portrayal components improves the IHO's ability to manage and extend the portrayal of products based on S-100. This Register supports wider use of registered items by making them publicly available and increasing their visibility to Portrayal Catalogue developers. This Part describes the content of the Portrayal Register.

2b-1.1 Conformance

This profile conforms to conformance class 2 of ISO 19106:2004.

2b-2 Normative references

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

ISO 19135:2005, *Geographic Information – Procedures for registration of items of geographic information*

ISO 19126:2009, *Geographic Information – Feature concept dictionaries and registers*

ISO 19117:2012, *Geographic Information – Portrayal*

2b-3 General concepts

2b-3.1 Register

As described in Part 2, a Register is simply a managed list. It is easier to maintain than a fixed document, because new items can be added as needed to the Register, and existing items in the Register can be clarified, superseded or retired. Each Register item has one or more dates associated with it that indicate when changes in its status occurred. This means that a Product Specification, defined at a given date, may reference an item in the Register at that specific point in time.

2b-3.2 Portrayal Register

The Portrayal Register specifies the independent sets of components used to build Portrayal Catalogues such as colours, symbols, and fonts. The Register also specifies supporting items such as context parameters, viewing groups, display planes and the components used within an optional Alerts Catalogue.

The Portrayal Register may be subdivided into different Domains; and may be used to develop a Portrayal Catalogue. Unlike a Portrayal Catalogue, the Portrayal Register does not define the portrayal rules or bind the portrayal or alerts/indications to a feature.

Such Registers of portrayal information may serve as sources of reference for similar Registers established by other geographic information communities as part of a system of cross-referencing.

2b-3.3 Portrayal catalogue

The Portrayal Catalogue contains portrayal rules that map the features to symbology; and optionally to alerts/indications. It also contains symbol definitions, colour definitions, portrayal parameters and portrayal management concepts. Portrayal Catalogues are described in detail in S-100 Part 9.

2b-4 IHO Portrayal Register

2b-4.1 Types of registered items

The following are types of items that may be registered:

1) Pixmap
2) Colour Value
3) Colour Token
4) Colour Palette
5) Symbol
6) Line Style
7) Area Fill
8) Font
9) Viewing Group
10) Viewing Group Layer
11) Display Mode
12) Display Plane
13) Context Parameter
14) Symbol Schema
15) Line Style Schema

16) Area Fill Schema
17) Pixmap Schema
18) Colour Profile Schema
19) Drawing Priority
20) Alert
21) Alert Message
22) Alert Highlight

2b-4.2 Data model of a Portrayal Register

2b-4.2.1 UML Model

Figures 2b-1 and 2b-2 show the register management and information models respectively of the hydrographic Portrayal Register:

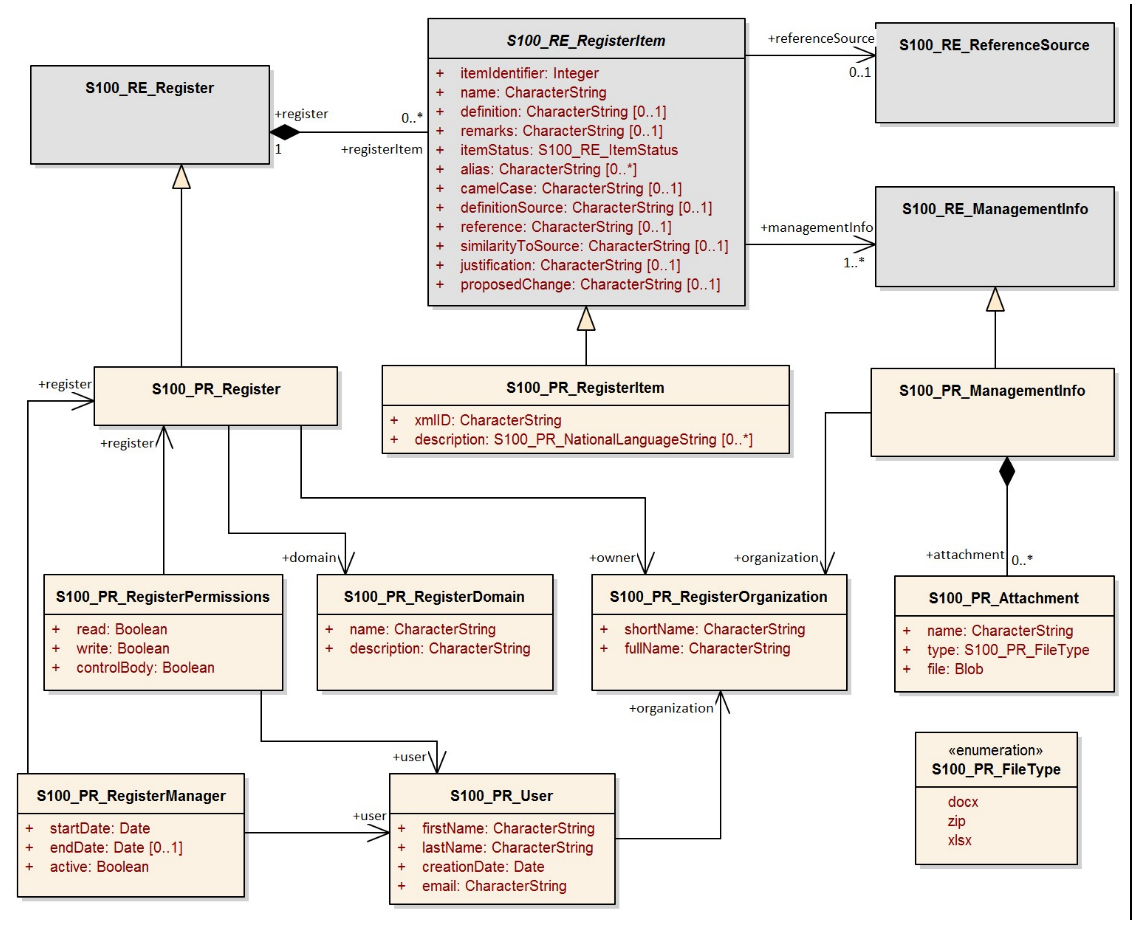


Figure 2b-1 – Portrayal Register management model

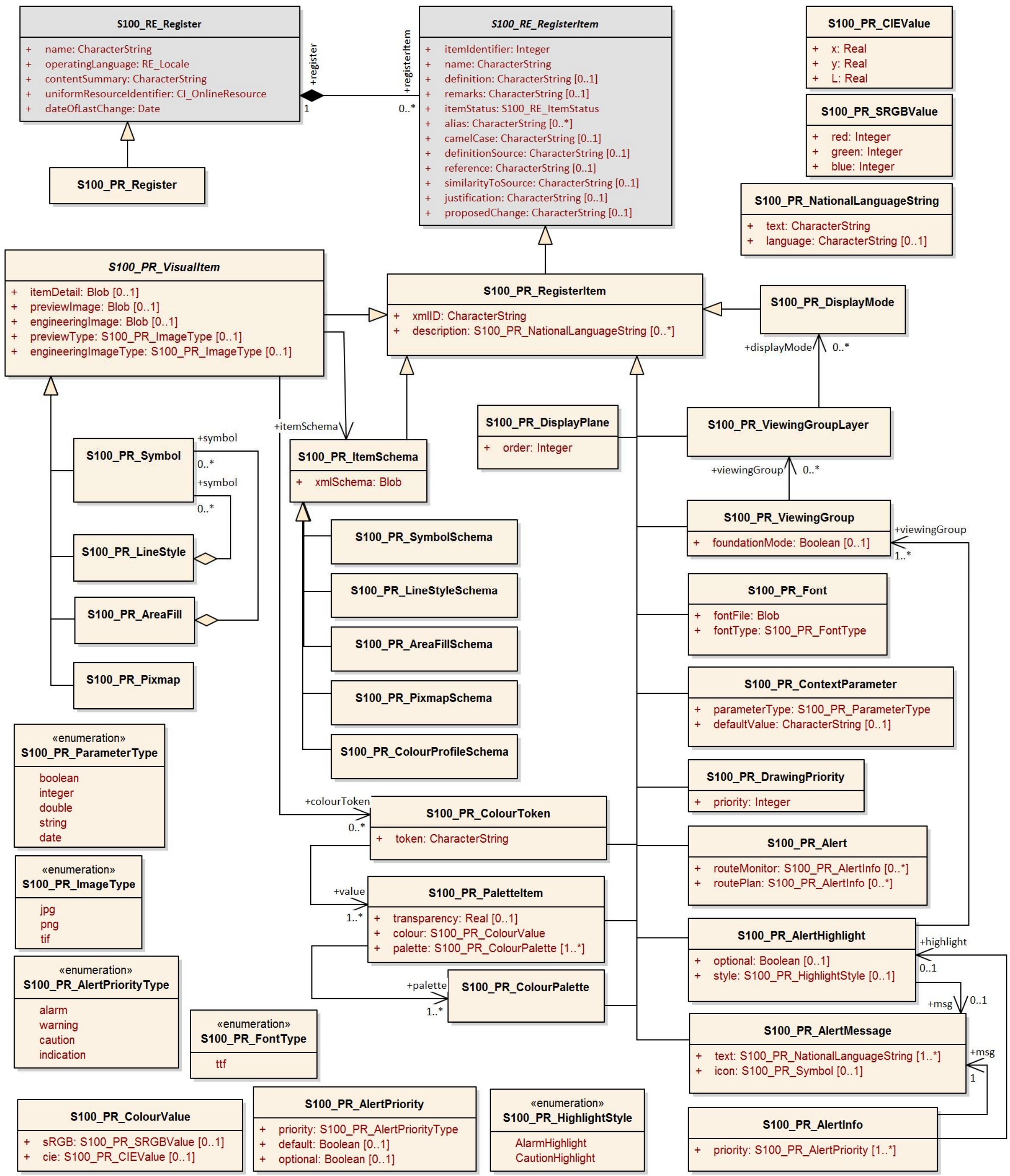


Figure 2b-2 – Portrayal Register information model

2b-4.2.2 S100_PR_Register

This class S100_PR_Register is a specialization of the class S100_RE_Register. It is extended with an 'owner' and 'domain'. An organization may have a dedicated Register and a Register is intended for a specific Domain.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_Register	Portrayal Register extension to S100_RE_Register	-	S100_RE_Register	Inherits all characteristics of S100_RE_RegisterItem and extended with domain and owner
Association	owner	The organization responsible for managing the contents of the Register	1	S100_PR_RegisterOrganization	
Association	domain	The domain for which the Register is primarily intended	1	S100_PR_RegisterDomain	

2b-4.2.3 S100_PR_RegisterDomain

This class indicates the Domain for which a Register is intended.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_RegisterDomain	Definition of a Domain	-	-	
Attribute	name	Name of Domain	1	CharacterString	
Attribute	description	Description of Domain	1	CharacterString	

2b-4.2.4 S100_PR_User

This class represents a user of the Register.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_RegisterUser	Identification of a Register user	-	-	
Attribute	firstName	First name of user	1	CharacterString	
Attribute	lastName	Last name of user	1	CharacterString	
Attribute	creationDate	Date user was entered	1	Date	
Attribute	email	Email of user	1	CharacterString	
Association	organization	Reference to organization	1	S100_PR_RegisterOrganization	

2b-4.2.5 S100_PR_RegisterManager

This class identifies a Register Manager along with current status and time period.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_RegisterManager	Indicates the manager of a Register	-	-	
Attribute	startDate	The start date of the manager	1	Date	
Attribute	endDate	The date on which the manager duties end	0..1	Date	
Attribute	active	Flag if manager is currently active	1	Boolean	
Association	register	The Register that the manager manages	1	S100_PR_Register	
Association	user	The user that is the manager	1	S100_PR_User	

2b-4.2.6 S100_PR_RegisterOrganization

This class represents a Register Organization.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_RegisterOrganization	Definition of a Register Organization	-	-	
Attribute	shortName	Abbreviated or simple form name	1	CharacterString	
Attribute	fullName	Full name of organization	1	CharacterString	

2b-4.2.7 S100_PR_RegisterPermissions

A class used to assign permissions to a Register user.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_RegisterPermissions	Register user permissions	-	-	
Attribute	read	Permission to read Register entries	1	Boolean	
Attribute	write	Permission to write Register entries	1	Boolean	
Attribute	controlBody	User is Control Body	1	Boolean	Part 2 Management of Registers
Association	register	The Register that the manager manages	1	S100_PR_Register	
Association	user	The user that is the manager	1	S100_PR_User	

2b-4.2.8 S100_PR_ ManagementInfo

This class is a portrayal extension of the S100_RE_ManagementInfo class with a reference to an Organization object and possible attachments.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_ManagementInfo	Extension of S100_RE_ManagementInfo	-	S100_RE_ManagementInfo	
Association	organization	Organization submitting Register entries	1	S100_PR_RegisterOrganization	
Association	attachment	Attached file or files	0..*	S100_PR_Attachment	

2b-4.2.9 S100_PR_ Attachment

This class handles attachments.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_ Attachment	Holds an attached file	-	-	
Attribute	name	Name of attachment	1	CharacterString	
Attribute	type	Type of attachment	1	S100_PR_FileType	
Attribute	file	The attachment	1	Blob	

2b-4.2.10 S100_PR_ RegisterItem

The class S100_PR_RegisterItem is a specialization of the class S100_RE_RegisterItem which carries a valid XML identifier to be used in a Portrayal Catalogue.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_RegisterItem	Extension of S100_RE_RegisterItem	-	S100_RE_RegisterItem	
Attribute	xmlID	Valid XML identifier string	1	CharacterString	
Attribute	description	Description in a national language	0..*	S100_PR_NationalLanguageString	Provides S100_RE_RegisterItem definition attribute in alternate languages, supporting population of Portrayal Catalogue language-independent descriptions

2b-4.2.11 S100_PR_ VisualItem

An abstract specialization of S100_PR_RegisterItem to represent 'symbol', 'lineStyle', 'areaFill' or 'pixmap'. The visual items each have an XML identifier string and XML document defining the item details as well as a preview image and an engineering image with dimensions.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_VisualItem	Abstract class representing a graphic element such as a symbol or linestyle	-	S100_PR_ RegisterItem	
Attribute	itemDetail	The XML file of the item	0..1	Blob	
Attribute	previewImage	A preview image of the item	0..1	Blob	
Attribute	engineeringImage	The engineering image with measurements	0..1	Blob	
Attribute	previewType	The file type of the preview image	0..1	S100_PR_ImageType	Required if previewImage is populated
Attribute	engineeringImageType	The file type of the engineering image	0..1	S100_PR_ImageType	Required if engineeringImage is populated
Association	itemSchema	The XML schema to validate the item	1	S100_PR_ItemSchema	
Association	colourToken	The colour tokens used by the visual item	0..*	S100_PR_ColourToken	Needed to identify dependencies when assembling a Portrayal Catalogue

2b-4.2.12 S100_PR_ Symbol

The class S100_PR_Symbol is a specialization of the class S100_PR_VisualItem used to register a symbol according to Part 9 Portrayal.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_Symbol	Symbol visual item	-	S100_PR_VisualItem	

2b-4.2.13 S100_PR_ LineStyle

The class S100_PR_LineStyle is a specialization of the class S100_PR_VisualItem used to register a linestyle according to Part 9 Portrayal.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_LineStyle	Line style visual item	-	S100_PR_VisualItem	
Association	symbol	Symbols used by the line style	0..*	S100_PR_ Symbol	Used to identify dependencies

2b-4.2.14 S100_PR_ AreaFill

The class S100_PR_AreaFill is a specialization of the class S100_PR_VisualItem used to register an area fill according to Part 9 Portrayal.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_AreaFill	Area fill visual item	-	S100_PR_VisualItem	
Association	symbol	Symbols used by the area fill	0..*	S100_PR_ Symbol	Used to identify dependencies

2b-4.2.15 S100_PR_ Pixmap

The class S100_PR_Pixmap is a specialization of the class S100_PR_VisualItem used to register a pixmap according to Part 9 Portrayal.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_Pixmap	Pixmap visual item	-	S100_PR_VisualItem	

2b-4.2.16 S100_PR_ ItemSchema

The class S100_PR_ItemSchema is a specialization of the class S100_PR_RegisterItem used to register a portrayal item schema according to Part 9 Portrayal.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_ItemSchema	Schema of an XML portrayal item	-	S100_PR_RegisterItem	
Attribute	xmlSchema	The XML schema stored as a Blob	1	Blob	Schema from Part 9 Portrayal

2b-4.2.16.1 S100_PR_ SymbolSchema

The class is a specialization of S100_PR_ItemSchema and is used to register the Part 9 Portrayal symbol schema.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_SymbolSchema	Symbol schema	-	S100_PR_ItemSchema	

2b-4.2.16.2 S100_PR_ LineStyleSchema

The class is a specialization of S100_PR_ItemSchema and is used to register the Part 9 Portrayal line style schema.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_LineStyleSchema	Line style schema	-	S100_PR_ItemSchema	

2b-4.2.16.3 S100_PR_ AreaFillSchema

The class is a specialization of S100_PR_ItemSchema and is used to register the Part 9 Portrayal area fill schema.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_AreaFillSchema	Area fill schema	-	S100_PR_ItemSchema	

2b-4.2.16.4 S100_PR_ PixmapSchema

The class is a specialization of S100_PR_ItemSchema and is used to register the Part 9 Portrayal pixmap schema.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_PixmapSchema	Pixmap schema	-	S100_PR_ItemSchema	

2b-4.2.16.5 S100_PR_ ColourProfileSchema

The class is a specialization of S100_PR_ItemSchema and is used to register the Part 9 colour profile schema.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_ColourProfileSchema	Colour profile schema	-	S100_PR_ItemSchema	

2b-4.2.17 S100_PR_ColourToken

The class S100_PR_ColourToken is a specialization of the class S100_PR_RegisterItem. A color token defines a single reference for multiple color values, each of which is associated with a single colour palette.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_ColourToken	Definition of a colour token	-	S100_PR_RegisterItem	
Attribute	token	Identifier of the colour value(s)	1	CharacterString	
Association	value	The colour(s) associated with this token	1..*	S100_PR_PaletteItem	Used to identify dependencies Multiplicity if intended for use by ECDIS is 3..*

2b-4.2.18 S100_PR_ColourPalette

The class S100_PR_ColourPalette is a specialization of the class S100_PR_RegisterItem.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_ColourPalette	Definition of a colour palette	-	S100_PR_RegisterItem	

2b-4.2.19 S100_PR_PaletteItem

The class S100_PR_PaletteItem is a specialization of the class S100_PR_RegisterItem. It provides a colour value for colour tokens within a colour palette.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_PaletteItem	Definition of a colour palette entry	-	S100_PR_RegisterItem	
Attribute	transparency	Transparency	0..1	Real	Range is [0, 1], default is 0 (opaque)
Attribute	colour	The colour value	1	S100_PR_ColourValue	
Association	palette	The palette(s) associated with this item	1..*	S100_PR_ColourPalette	Used to identify dependencies

2b-4.2.20 S100_PR_DisplayMode

This is a specialization of the class S100_PR_RegisterItem used to register a Display Mode according to Part 9 Portrayal.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_DisplayMode	Used to register a Display Mode	-	S100_PR_RegisterItem	See Part 9 Portrayal

2b-4.2.21 S100_PR_ViewingGroupLayer

This is a specialization of the class S100_PR_RegisterItem used to register a Viewing Group Layer according to Part 9 Portrayal.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_ViewingGroupLayer	Used to register a Viewing Group Layer	-	S100_PR_RegisterItem	See Part 9 Portrayal
Association	displayMode	A display mode which may include this layer	0..*	S100_PR_DisplayMode	Used to identify dependencies

2b-4.2.22 S100_PR_ViewingGroup

This is a specialization of the class S100_PR_RegisterItem used to register a Viewing Group according to Part 9 Portrayal.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_ViewingGroup	Used to register a Viewing Group	-	S100_PR_RegisterItem	See Part 9 Portrayal

Attribute	foundationMode	Indicates the viewing group should always be on.	0..1	Boolean	Default is false
Association	viewingGroupLayer	A viewing group layer which may include this viewing group	0..*	S100_PR_ViewingGroupLayer	Used to identify dependencies

2b-4.2.23 S100_PR_DisplayPlane

This is a specialization of the class S100_PR_RegisterItem.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_DisplayPlane	The specific content for a display plane definition as a register item of type 'displayPlane'	-	S100_PR_RegisterItem	See Part 9 Portrayal
Attribute	order	Used to sort the drawing order of display planes	1	Integer	

2b-4.2.24 S100_PR_Font

This is a specialization of S100_PR_RegisterItem. Used to register a font file for use in a Portrayal Catalogue.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_Font	The specific content for a font file definition as a register item of type 'font'	-	S100_PR_RegisterItem	See Part 9 Portrayal
Attribute	fontFile	A font file for inclusion in a portrayal catalogue	1	Blob	
Attribute	fontType	The type of font file	1	S100_PR_FontType	Initially restricted to True Type Font

2b-4.2.25 S100_PR_DrawingPriority

S100_PR_DrawingPriority is a specialization of S100_PR_RegisterItem.

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_PR_DrawingPriority	Used to register drawing priorities to be used in a Portrayal Catalogue	-	S100_PR_ RegisterItem	See Part 9 Portrayal
Attribute	priority	Used to sort drawing instructions within a display plane	1	Integer	

2b-4.2.26 S100_PR_ContextParameter

S100_PR_ContextParameter is a specialization of S100_PR_RegisterItem.

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_PR_ContextParameter	The specific content for a context parameter as a register item of type 'contextParameter'	-	S100_PR_ RegisterItem	See Part 9 Portrayal
Attribute	parameterType	Data type of context parameter	1	S100_PR_ParameterType	
Attribute	defaultValue	Default or initial value	0..1	CharacterString	

2b-4.2.27 S100_PR_ColourValue

Identifies a colour value in one or more colour spaces.

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_PR_ColourValue	Describes a colour value in one or more colour spaces	-	-	At least one colour value must be present
Attribute	sRGB	The colour value in the sRGB colour space	0..1	S100_PR_SRGBValue	
Attribute	cie	The colour value in the CIE colour space	0..1	S100_PR_CIEValue	

2b-4.2.28 S100_PR_SRGBValue

Identifies a colour value in the sRGB colour space.

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_PR_SRGBValue	A colour value in the sRGB colour space	-	-	
Attribute	red	Amount of red	1	Integer	Range is [0,255]
Attribute	green	Amount of green	1	Integer	Range is [0,255]
Attribute	blue	Amount of blue	1	Integer	Range is [0,255]

2b-4.2.29 S100_PR_CIEValue

Identifies a colour value in the CIE colour space.

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_PR_CIEValue	A colour value in the CIE colour space	-	-	
Attribute	x	x component of the chromaticity	1	Real	Range [0,1]
Attribute	y	y component of the chromaticity	1	Real	Range [0,1]

Attribute	L	Luminance in cd/m2	1	Real	

2b-4.2.30 S100_PR_Alert

S100_PR_Alert is a specialization of S100_PR_RegisterItem. An alert is associated with features by Portrayal Catalogue rules.

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_PR_Alert	Provides a priority, message, and the viewing groups associated with a graphical highlight.	-	S100_PR_RegisterItem	One of routeMonitor or routePlan must be present
Attribute	routeMonitor	Describes the attributes of the alert during route monitoring	0..*	S100_PR_AlertInfo	
Attribute	routePlan	Describes the attributes of the alert during route planning	0..*	S100_PR_AlertInfo	

2b-4.2.31 S100_PR_AlertInfo

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_PR_AlertInfo	The attributes of an alert	-	-	
Attribute	priority	The priority of the alert	1..*	S100_PR_AlertPriority	Multiplicity supports user selection of priority
Association	msg	A message to display while the alert is active	1	S100_PR_AlertMessage	
Association	highlight	Supports toggling display of the graphical highlight	0..1	S100_PR_AlertHighlight	

2b-4.2.32 S100_PR_AlertPriority

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_PR_AlertPriority	The priority of an alert	-	-	
Attribute	priority	The priority value	1	S100_PR_AlertPriorityType	
Attribute	default	Indicates this priority is the default	0..1	Boolean	Default is false
Attribute	optional	Indicates exposing this priority to the user is optional	0..1	Boolean	Default is false

2b-4.2.33 S100_PR_AlertMessage

S100_PR_AlertMessage is a specialization of S100_PR_RegisterItem. Messages are displayed while any associated alert is active.

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_PR_AlertMessage	A message associated with one or more alerts	-	S100_PR_RegisterItem	
Attribute	text	Message in one or more languages	1..*	S100_PR_NationalLanguageString	Message in English must be provided
Association	icon	A symbol to display with or instead of the text.	0..1	S100_PR_Symbol	

2b-4.2.34 S100_PR_AlertHighlight

S100_PR_AlertHighlight is a specialization of S100_PR_RegisterItem. Highlights associate viewing groups with alert information, which allows users to disable graphical highlighting of one or more active S100_PR_Alerts.

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_PR_AlertHighlight	Associates display parameters with alerts	-	S100_PR_RegisterItem	
Attribute	style	Overrides the style of graphical highlight shown in the chart area	0..1	S100_PR_HighlightStyle	When empty, style is inferred from the alert priority • Alarm: AlarmHighlight • Others: CautionHighlight
Attribute	optional	Indicates whether allowing the highlight to be turned off is optional	0..1	Boolean	Default is false
Association	viewingGroup	The viewing group(s) associated with an alert highlight (Portrayal Catalogue rules generate the drawing instructions which implement the highlight)	1..*	S100_PR_ViewingGroup	Multiple viewing groups are present to allow highlighting of alerts triggered by specific feature types to be disabled independently For example, ProhibitedAreaHighlight supports disabling highlighting of Seaplane Landing Areas while other prohibited areas continue to be highlighted When multiple viewing groups are provided, consideration should be given to providing a viewing group layer to toggle all the viewing groups at once
Association	msg	A message to display while any of the viewing groups are disabled	0..1	S100_PR_AlertMessage	

2b-4.2.35 S100_PR_NationalLanguageString

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_PR_NationalLanguageString	Text specific to a national language	-	-	
Attribute	text	A string in a national language	1	CharacterString	
Attribute	language	ISO 639-2/T 3-letter language code	0..1	CharacterString	Default is “eng“

2b-4.2.36 S100_PR_FileType

Item	Name	Description	Code	Remarks
Enumeration	S100_PR_FileType	The type and format of a file.	-	
Value	docx	Office Open XML Document	-	Zip and XML-based file format for documents. Not to be confused with OpenOffice format or generic XML
Value	zip	Zip archive format	-	
Value	xlsx	Office Open XML Workbook	-	Zip and XML-based file format for spreadsheets. Not to be confused with OpenOffice format or generic XML

2b-4.2.37 S100_PR_FontType

Item	Name	Description	Code	Remarks
Enumeration	S100_PR_FontType	A font specification	-	
Value	ttf	TrueType font	-	

2b-4.2.38 S100_PR_ImageType

Item	Name	Description	Code	Remarks
Enumeration	S100_PR_ImageType	An image specification	-	
Value	jpg	JPEG 2000 image coding system	-	
Value	png	Portable Network Graphics format	-	
Value	tif	Tagged Image File Format	-	

2b-4.2.39 S100_PR_AlertPriorityType

Item	Name	Description	Code	Remarks
Enumeration	S100_PR_AlertPriorityType	Defines the possible priorities for an alert.	-	
Value	alarm	Indicates conditions requiring immediate attention and action by the bridge team	-	From MSC.252(83) 19.1.2
Value	warning	Indicates changed conditions and should be presented for precautionary reasons which are not immediately hazardous but which may become so, if no action is taken	-	From MSC.252(83) 19.1.3
Value	caution	Indicates a condition which does not warrant an alarm or warning condition, but still requires attention and out of the ordinary consideration of the situation or of given information	-	From MSC.252(83) 19.1.4
Value	indication	Display of regular information and conditions	-	From MSC.252(83) appendix 1

2b-4.2.40 S100_PR_ParameterType

The definition and members of enumeration S100_PR_ParameterType are the same as ParameterType in Part 9, clause 9-13.3.30.

2b-4.2.41 S100_PR_HighlightStyle

Item	Name	Description	Code	Remarks
Enumeration	S100_PR_HighlightStyle	Defines the possible graphical highlight styles for an alert	-	
Value	AlarmHighlight	Red highlight	-	IEC 62288:2014 Table A.3 Navigation symbols, entry 3.5 b
Value	CautionHighlight	Yellow highlight	-	IEC 62288:2014 Table A.3 Navigation symbols, entry 3.5 c

Page intentionally left blank

S-100 – Part 3

General Feature Model and Rules for Application Schema

Page intentionally left blank

Contents

3-1 Scope

This Part introduces a General Feature Model (GFM) which is a conceptual model of features, their characteristics and associations. It also describes the rules for developing an Application Schema which is a basic part of any S-100 based Product Specification.

The scope of this Part includes:

1) Conceptual modelling of features and their properties from a reality;
2) Conceptual modelling of information types and their attributes;
3) Definition of Application Schema;
4) Rules for Application Schema;

The following is outside scope:

1) Representation of feature types and their properties and information types and their properties in a Catalogue;
2) Representation of metadata;
3) Rules for mapping one Application Schema to another;
4) Implementation of the Application Schema in a computer environment;
5) Computer system and Application Schema software design;
6) Programming.

Computer systems, software design and programming are not addressed in this document.

3-2 Conformance

This profile conforms to conformance class 2 of ISO 19106:2004. The following is a brief description of the specializations and generalizations where the S-100 General Feature Model differs from ISO 19109.

1) A new S100_GF_NamedType is introduced.
2) A new S100_GF_ObjectType is introduced as a specialisation of S100_GF_NamedType.
3) A new S100_GF_InformationType is introduced as a specialisation of S100_GF_ObjectType, it is constrained to associations with S100_GF_ThematicAttributeType.
4) S100_GF_FeatureType is a specialization of S100_GF_ObjectType.
5) S100_GF_AttributeType is a specialization of GF_AttributeType in that it is abstract in S-100.
6) A new abstract S100_GF_SimpleAttributeType is introduced as a specialisation of S100_GF_ThematicAttributeType.
7) GF_Operation is not used.
8) GF_InheritanceRelation is not used; feature inheritance is represented by the association inheritance.
9) The association attributeOfAttribute is not used. The concept of the complex attribute is used in S-100 to perform a similar function.
10) S100_GF_AssociationType does not use the generalization association between GF_AssociationType and GF_FeatureType. Instead it is a specialisation of S100_GF_NamedType.
11) S100_GF_AssociationType is associated with S100_GF_ThematicAttributeType by a UML aggregation relationship. This means associations can have descriptive characteristics.

12) New metaclasses S100_GF_FeatureAssociationType and S100_GF_InformationAssociationType are introduced as specialisations of S100_GF_AssociationType.

13) The association role linkBetween of the GF_FeatureType/GF_AssociationType relationship in ISO 19109 is realized as follows:

    a) Role linkBetween of the S100_FeatureType/S100_GF_FeatureAssociationType relationship;

    b) Role linkBetween of the S100_InformationType / S100_GF_InformationAssociationType relationship;

    c) Role informationLink of the S100_ObjectType / S100_InformationAssociationType relationship.

    This means that associations that include only feature types have semantics and multiplicity constraints that are different from associations that include at least one information type.

14) GF_LocationAttributeType, GF_TemporalAttributeType, GF_MetaDataAttributeType and GF_QualityAttributeType are not used.

Further reference or explanation of the above changes can be found in the following text where appropriate.

3-3 References

ISO 8601:2004, *Data elements and interchange formats – Information interchange – Representation of dates and times*

ISO 19106:2003, *Geographic information - Geographic Information – Profiles*

ISO 19108:2002, *Geographical Information – Temporal Schema* (as corrected by Technical Corrigendum 1 – 2006)

ISO 19107:2003, *Geographic information - Spatial schema*

ISO 19109:2005, *Geographic information - Rules for application schema*

ISO 19110:2005, *Geographic information - Methodology for feature cataloguing*

ISO 19115-1:2018, *Geographic information – Metadata – Part 1 – Fundamentals* (as updated by Amendment 1, 2018)

ISO/CD 19115-2, *Geographic information - Metadata - Part 2 – Extensions for imagery and gridded data*

3-4 Context

3-4.1 Objects

The data content of a geographic application is defined in accordance with a view of real world features and in the context of the requirements of a particular application. The content is structured in terms of objects. This document considers two types of object:

1) Features – features are defined together with their properties.
2) Information Types – information types are used to share information among features and other information types. Information types have only thematic attribute properties.

The GFM provides a conceptual model for these objects. The definitions for object types are held in a Feature Catalogue. The GFM also acts as a conceptual model for the Feature Catalogue.

3-4.2 Derivation of the General Feature Model

A conceptual model of types that shall be used in S-100 products is presented in this document. It is known as the GFM and is derived from the ISO 19109 General Feature Model by realization of its classes (Figure 3-1).

3-5 Principles for defining features and information types

3-5.1 Identifiable objects

3-5.1.1 Features

A feature is an abstract representation of real world phenomenon. Features have two aspects – feature type and feature instance. A feature type is a class and is defined in a Feature Catalogue. A feature instance is a single occurrence of the feature type and represented as an object in a data set.

3-5.1.2 Information types

An information type is a class of object which is defined in a Feature Catalogue. An instance of an information type is an identifiable unit of information in a data set. Information types have only thematic attribute properties. An instance of an information type may be associated with one or more feature instances or other instances of information type.

EXAMPLE A chart note may be modelled as an information type

3-5.2 The General Feature Model

3-5.2.1 Introduction

This sub-clause identifies and describes the concepts used to define features and information types and their relationships. These concepts are expressed in a conceptual model called the GFM.

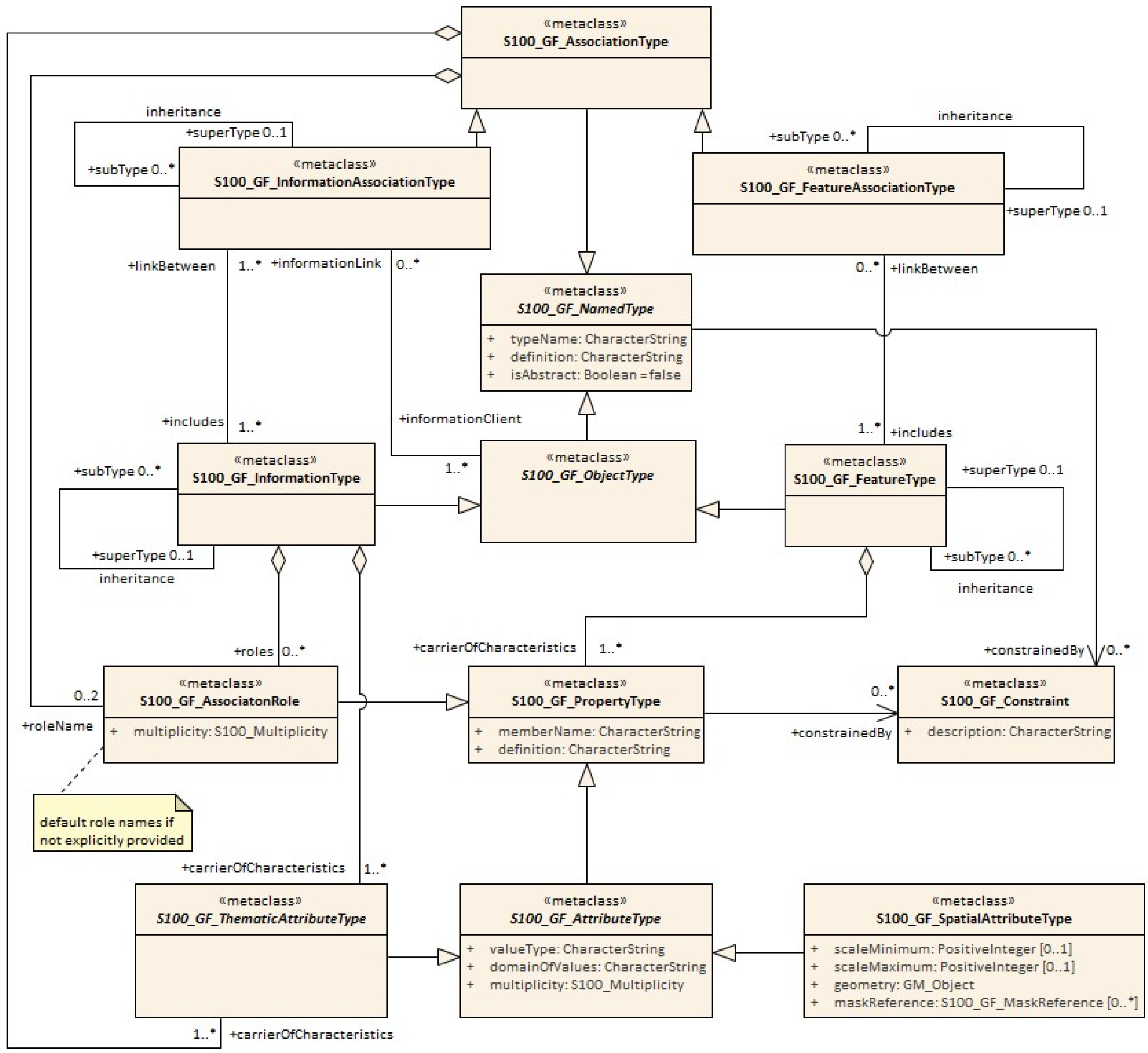


Figure 3-1 – The General Feature Model

3-5.2.2 The purpose of the GFM

The GFM is a basis for the classification of features and information types and their properties. The GFM also acts as the basis for the structure of Feature Catalogues.

3-5.2.3 The main structure of the GFM

Figure 3-1 shows a UML model of the S-100 GFM.

The following clauses define the elements of the GFM.

3-5.2.4 S100_GF_NamedType

The class S100_GF_NamedType is not realized from ISO 19109 but is introduced specifically for the S-100 GFM. It is an abstract super-class of the classes S100_GF_ObjectType and S100_GF_AssociationType. The intention in introducing this class is to show the commonality between object types and association types within S-100. Both types are core identifiable objects of S-100 data Schemas.

Table 3-1— S100_GF_NamedType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_NamedType	Abstract base class for object types and association types within the GFM	-	-
Attribute	typeName	Name of the named type. The name shall be unique within a namespace	1	CharacterString
Attribute	definition	Definition that describes the named type	1	CharacterString
Attribute	isAbstract	If true, the named type acts as an abstract supertype. It is not possible to create an instance of an abstract type	1	Boolean
Role	constrainedBy	The role specifies that a constraint is made on the named type	0..*	S100_GF_Constraint

3-5.2.5 S100_GF_ObjectType

The class S100_GF_ObjectType is not realized from ISO 19109 but is introduced specifically for the S-100 GFM. It is an abstract super-class of the classes S100_GF_FeatureType and S100_GF_InformationType. The intention in introducing this class is to show the commonality between feature types and information types in particular the ability of these classes to be linked to information types by means of an information association.

Table 3-2— S100_GF_ObjectType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_ObjectType	Abstract base class for object types within the GFM	-	-
Role	informationLink	Link to an information association that describes the relationship to an instance of an information type	0..*	S100_GF_Information AssociationType

3-5.2.6 S100_GF_FeatureType

The class S100_GF_FeatureType is a realization of the ISO 19109 class GF_FeatureType. It differs from the ISO class in the following ways:

- It is a sub-type of the class S100_GF_NamedType;
- It does not realize the Generalization and Specialization associations with the class GF_InheritanceRelation. Instead, the class has an association with itself with the roles subType and superType. GF_InheritanceRelation is not realized in the S-100 GFM;
- The multiplicity of the superType is 0..1 to represent the concept that a feature may have a maximum of one superType. This is in order to prevent multiple-inheritance in S-100;
- The multiplicity of the role carrierOfCharacteristics with S100_GF_PropertyType (the S-100 realization of GF_PropertyType) is changed from 0..* to 1..*. An S-100 feature must have properties.

Table 3-3— S100_GF_FeatureType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_FeatureType	A type for an abstract representation of a real world phenomenon	-	-
Role	superType	The more generic feature type from which this feature type is derived	0..1	S100_GF_Feature Type
Role	subType	The more specific feature types which are derived from this feature type	0..*	S100_GF_Feature Type
Role	linkBetween	A link to a feature association that specify the relationship between one feature type and the same or another feature type	0..*	S100_GF_Feature AssociationType
Role	carrierOfCharacteristics	Attributes and roles that describe the characteristics of a feature type	1..*	S100_GF_Property Type

3-5.2.7 S100_GF_PropertyType

The class S100_GF_PropertyType is a realization of the ISO 19109 class GF_PropertyType. It differs from the ISO class in the following ways:

1) The multiplicity of the association with S100_GF_FeatureType is changed from 1 to 1..*. This change represents the way that features and properties are described in the S-100 Feature Catalogue. Property type definitions can be used in one or more feature type definitions;
2) The association type of the association with S100_GF_FeatureType is changed from composition to aggregation as a result of the change in multiplicity described above.

Table 3-4 — S100_GF_PropertyType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_PropertyType	Abstract base class for all properties of a feature type. These are attributes and roles	-	-
Attribute	memberName	Name of the attribute or role	1	CharacterString
Attribute	definition	Description of the attribute or role of the feature type	1	CharacterString
Role	constrainedBy	The role specifies that a constraint is made on the property	0..*	S100_GF_Constraint

3-5.2.8 S100_GF_AttributeType

The class S100_GF_AttributeType is the S-100 realization of GF_AttributeType. It is largely identical to the ISO 19109 class but differs in the following way:

1) The association attributeOfAttribute is not realized in the S-100 GFM. S-100 introduces, instead, the concept of complex attributes. Complex attributes are described further in ISO 19109 subclause 7.4.

Table 3-5— S100_GF_AttributeType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_AttributeType	Abstract base class for all attributes of feature types. In this model are two sub classes: thematic attributes and spatial attributes	-	-
Attribute	valueType	The data type of the attribute value	1	CharacterString
Attribute	domainOfValues	Description of a set of values. For codelist types this may be a URI identifying a dictionary or “vocabulary”	1	CharacterString
Attribute	multiplicity	The number of instances of the attribute that may be associated with a single instance of a feature type	1	S100_Multiplicity

3-5.2.9 S100_GF_AssociationRole

The class S100_GF_AssociationRole is the S-100 realization of the ISO 19109 class GF_AssociationRole.

Table 3-6 — S100_GF_AssociationRole

Role Name	Name	Description	Mult.	Type
Class	S100_GF_ AssociationRole	A role used in an association	-	-
Attribute	multiplicity	The number of objects that may be associated within the association	1	S100_Multiplicity

3-5.2.10 GF_Operation

The class GF_Operation is not realized in the S-100 GFM because S-100 supports only the data transfer model. Datasets cannot contain operations.

3-5.2.11 S100_GF_AssociationType

The class S100_GF_AssociationType is the S-100 realization of the ISO 19109 class GF_AssociationType. It differs from the ISO 19109 class in the following way:

1) The ISO 19109 GFM models GF_AssociationType as a subtype of the class GF_FeatureType. This is done for reasons which are set out in Note 1 of ISO 19109 clause 7.3.9. The S-100 model does not model the class as a subtype of S100_GF_FeatureType. Within S-100 associations between feature types are not considered abstractions of real world phenomena. The result of this approach to modelling the GFM is that the only properties associations can have are thematic attributes.
2) The multiplicity of roleName is 0..2 instead of 1..*. The lower bound of 0 means the role is one of the default roles “source” or “target” and this is obvious from the Application Schema’s semantics of the association type’s name and the names of the participating feature or information classes. The upper bound expresses the constraint that S-100 does not allow associations with more than two participating classes.

Table 3-7— S100_GF_AssociationType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_AssociationType	Abstract base class for feature associations and information associations	-	-
Role	carrierOfCharacteristics	The thematic attributes that describes the association	0..*	S100_GF_ThematicAttributeType
Role	roleName	The roles that describes the ends of the association	0..2	S100_GF_AssociationRole

3-5.2.12 S100_GF_InformationType

S100_GF_InformationType is the class for information types within S-100. An information type is an identifiable object that can be associated with features in order to carry information particular to the associated features. An example of an information type might be a Chart Note. Information types can also be associated with each other. This could be done where there is further supplementary information that is relevant to the information type or where there is a need to translate the information. For example a primary information object carrying a Chart Note may contain text in English and an associated supplementary information object may carry the same text in German.

The characteristics of information types shall be carried by thematic attribute types only. Therefore, S100_GF_InformationType is associated with only S100_GF_ThematicAttributeType rather than the more generic class S100_GF_PropertyType. The associations to information types are modelled by means of the type S100_InformationAssociationType.

Table 3-8 — S100_GF_InformationType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_Information Type	A type for an identifiable object carrying supplementary information for other objects	-	-
Role	superType	The more generic information type from which this information type is derived	0..1	S100_GF_ InformationType
Role	subType	The more specific information types which are derived from this information type	0..*	S100_GF_ InformationType
Role	linkBetween	A link to an information association that specifies the relationship between one object type and this information type	0..*	S100_GF_Information AssociationType
Role	carrierOfCharacteristics	Thematic attributes that describe the characteristics of an information type	1..*	S100_GF_Thematic AttributeType
Role	roles	Roles for associations to other information type that supplying supplementary information	0..*	S100_GF_AssociationRole

3-5.2.13 S100_GF_FeatureAssociationType

The class S100_GF_FeatureAssociationType is not realized from ISO 19109 but is introduced specifically for the S-100 GFM. The reason for this is that in S-100 two types of associations are distinguished: feature associations and information associations. They are both semantically different and different in the model. This class describes the feature association. A feature association is the description of the relationship between two instances of feature types. It can be characterized by thematic attributes and has normally two roles. The roles describe the ends of the relationship since such relationship is usually not symmetric.

Table 3-9— S100_GF_FeatureAssociationType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_Feature AssociationType	A class for the description of a relationship between two feature types	-	-
Role	superType	The more generic feature association from which this feature association is derived	0..1	S100_GF_Feature AssociationType
Role	subType	The more specific feature associations which are derived from this feature association	0..*	S100_GF_Feature AssociationType
Role	includes	The feature types which are included in this relationship	1..*	S100_GF_FeatureType

3-5.2.14 S100_GF_InformationAssociationType

The class S100_GF_InformationAssociationType is not realized from ISO 19109 but is introduced specifically for the S-100 GFM. The reason for this is that in S-100 two types of associations are distinguished: feature associations and information associations. They are both semantically different and different in the model. This class describes the information association. An information association is the description of the relationship between an arbitrary object and an information type that supplies additional information for that object. The relationship can be characterized by thematic attributes and a role.

Table 3-10— S100_GF_InformationAssociationType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_Information AssociationType	A class for the description of a relationship between an object and an information type	-	-
Role	superType	The more generic information association from which this information association is derived	0..1	S100_GF_ InformationAssociationType
Role	subType	The more specific feature associations which are derived from this feature association	0..*	S100_GF_ InformationAssociationType
Role	includes	The information type that is included in the relationship	1..*	S100_GF_InformationType
Role	informationClient	The object types that act as client in the information association	1..*	S100_GF_ObjectType

3-5.2.15 S100_GF_Constraint

The class S100_GF_Constraint is a realization of the ISO 19109 class GF_Constraint with an association to S100_GF_NamedType instead of the ISO 19109 association to GF_Feature_Type.

Table 3-11— S100_GF_Constraint

Role Name	Name	Description	Mult.	Type
Class	S100_GF_ Constraint	Class for constraints that may be associated with named types or their properties	-	-
Attribute	description	The constraint described in natural language and/or in formal notation	1	CharacterString

3-5.3 Attributes of feature types

3-5.3.1 Introduction

This clause describes in more detail the role of attributes of features and information types.

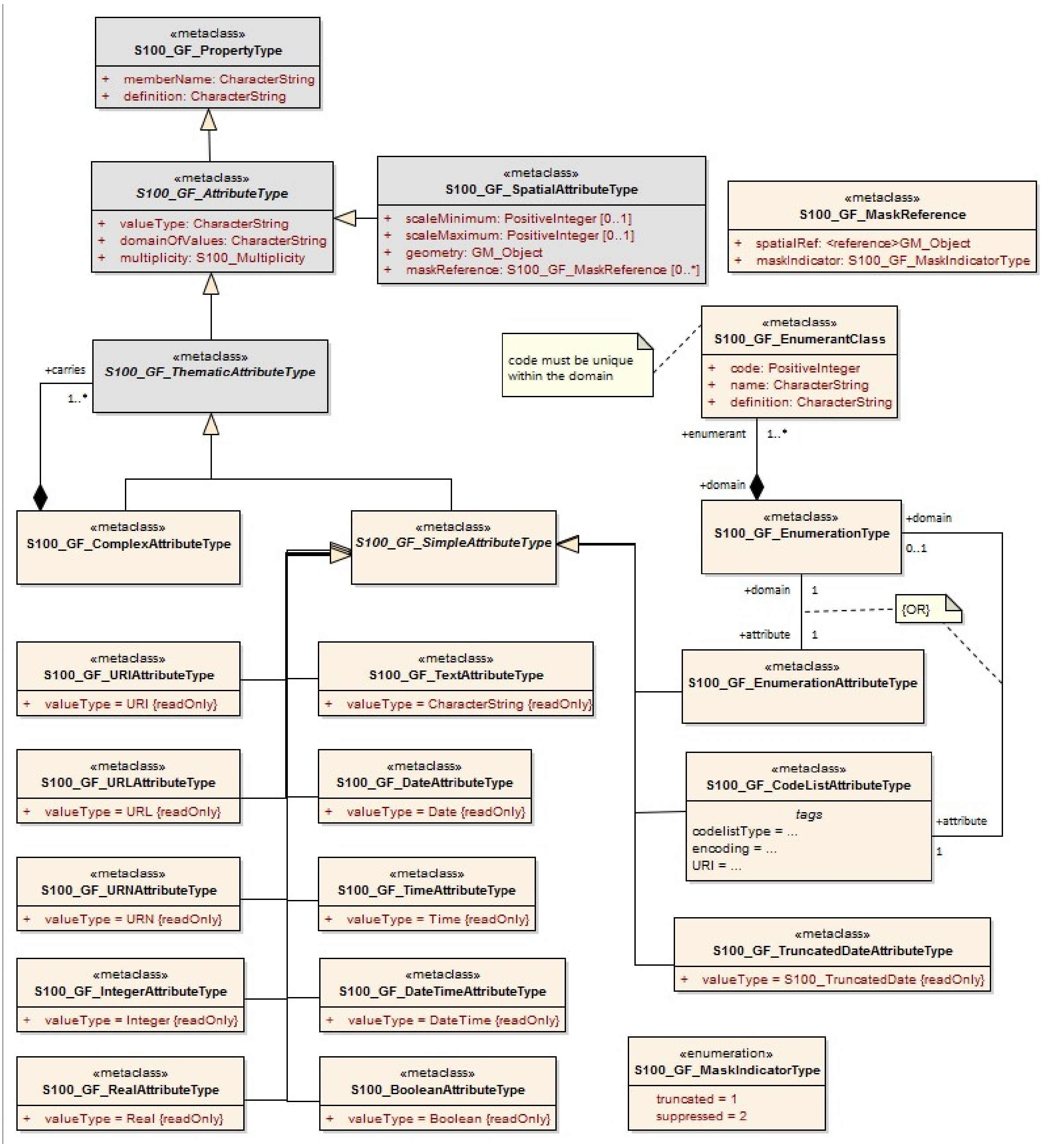


Figure 3-2 — Attributes

3-5.3.2 S100_GF_ThematicAttributeType

The class S100_GF_ThematicAttributeType is a realization of the ISO 19109 class GF_ThematicAttributeType. Thematic attribute types carry descriptive characteristics of objects other than those specified in ISO 19109 clauses 7.4.3 – 7.4.7. This class differs from the ISO 19109 class in the following ways:

1) GF_ThematicAttributeType is defined in ISO 19109 as a concrete class. The S-100 GFM realization is an abstract class with two concrete subclasses – S100_GF_SimpleAttributeType and S100_GF_ComplexAttributeType.

2) Temporal information shall have their value type defined by the types Date, Time, DateTime, S100_TruncatedDate or complex structures using combinations of the primitive temporal types.

Table 3-12— S100_GF_ThematicAttributeType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_ ThematicAttributeType	Abstract base class for all attributes other than spatial attributes	-	-

3-5.3.3 S100_GF_ComplexAttributeType

The class S100_GF_ComplexAttributeType is introduced in the S-100 GFM. Complex attributes are a composition of other attributes either simple or complex.

3-5.3.4 S100_GF_SimpleAttributeType

The class S100_GF_SimpleAttributeType is introduced in the S-100 GFM. A simple attribute type carries a descriptive characteristic of a named type.

3-5.3.5 S100_GF_SpatialAttributeType

The class S100_GF_SpatialAttributeType is a realization of the ISO 19109 class GF_SpatialAttributeType. A spatial attribute type shall have a GM_Object as its value type. GM_Object and its sub-types are defined in the Spatial Schema, S-100 Part 7.

Table 3-13— S100_GF_SpatialAttributeType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_ SpatialAttributeType	Class representing a spatial attribute, which shall be used to express spatial characteristics of a feature type	-	-
Attribute	scaleMinimum	The denominator of the smallest scale for which the referenced geometry can be used for the instance of the feature type (for example, for depiction)	0..1	PositiveInteger
Attribute	scaleMaximum	The denominator of the largest scale for which the referenced geometry can be used for the instance of the feature type (for example, for depiction)	0..1	PositiveInteger
Attribute	geometry	The object that describes the geometry of an instance of a feature type	1	GM_Object
Attribute	maskReference	Reference indicating masked or truncated spatial primitives or objects	0..*	S100_MaskReference

Masking or truncation shall be indicated by providing the identifiers of the masked or truncated primitives and an indicator of whether the referenced primitive is masked or truncated in *maskReference* attributes. The structure of the *maskReference* attribute is defined by the type *S100_GF_MaskReference*, shown in Table 3-14 below.

The implementation of mask references in different S-100 formats is specified in the respective data format specifications (Part 10a for the ISO 8211 data format and Part 10b for the GML format) and may use constructs built into the core specification. For example, the S-100 GML format uses the GML type *ReferenceType* with restrictions on allowed values of the *xlink:href* and *xlink:role* attributes; the ISO 8211 format uses unsigned integers containing the record identifier of a spatial object and the numeric code of the mask indicator value.

The spatial objects referenced in the masking attribute must be among the components of the GM_Object that constitutes the spatial object referenced by the same instance of the spatial attribute. They may be components at any level, for example, components of components, etc. (In other words, the masked or truncated geometry must be part of the geometry of that particular instance of the spatial attribute.)

Product Specifications should restrict the use of masking to specific spatial types if needed, for example, to curves.

Table 3-14— S100_GF_MaskReference

Role Name	Name	Description	Mult.	Type
Class	S100_GF_MaskReference	Reference to a masked or truncated spatial primitive. Model is based on gml:Reference but limits the allowed attributes and makes the identifier and role mandatory	-	-
Attribute	spatialRef (alias xlink:href)	Identifier of a spatial primitive	1	<(reference>GM_Object
Attribute	maskIndicator (alias xlink:role)	Indicates whether a spatial primitive is masked or truncated by the dataset limit	1	Enumeration *S100_GF_MaskIndicatorType*

Table 3-15— S100_GF_MaskIndicatorType

Item	Name	Description	Code	Remarks
Enumeration	S100_GF_MaskIndicatorType	Indicates masking or truncation at the dataset limit	-	-
Literal	truncated	The spatial primitive is truncated at the dataset limit	1	
Literal	suppressed	Portrayal of the spatial primitive is suppressed	2	

3-5.3.6 GF_TemporalAttributeType

The ISO 19109 class GF_TemporalAttributeType is not realized explicitly in the S-100 GFM. Temporal information shall be modelled using the thematic attribute type S100_GF_ThematicAttributeType (see clause 3-6.4.4 for more details).

3-5.3.7 GF_MetadataAttributeType

The ISO 19109 class GF_MetadataAttributeType is not realized explicitly in the S-100 GFM. Metadata types shall be modelled using complex thematic attributes which realize types from the S-100 Part 4a metadata component. The complex thematic attributes shall be defined in a Feature Catalogue.

3-5.3.8 GF_QualityAttributeType

The ISO 19109 class GF_QualityAttributeType is not realized explicitly in the S-100 GFM. Quality metadata types shall be modelled using complex thematic attributes which realize types from the S-100 Part 4c Appendix 4c-A Data Quality. The complex thematic attributes shall be defined in a Feature Catalogue.

3-5.3.9 GF_LocationAttributeType

The ISO 19109 class GF_LocationAttributeType is not realized in the S-100 GFM.

3-5.3.10 S100_TruncatedDateAttributeType

The class S100_TruncatedDateAttributeType is intended for modelling date values with one or more of the more significant components omitted. This allows partial dates to be used, for example, for recurring periods.

3-5.3.11 S100_GF_CodelistAttributeType

The class S100_GF_CodelistAttributeType is introduced in the S-100 GFM for modelling S-100 codelists. Codelist attributes must be associated to either an enumeration (for open enumeration codelists) or a dictionary (for open and closed dictionary codelists) but not both. The structure of the dictionary is defined by an external specification.

Table 3-16— S100_GF_CodelistAttributeType

Role Name	Name	Description	Mult.	Type	Remarks
Class	S100_GF_ CodelistAttributeType	Abstract base class for S100_Codelist attributes	-	-	-
Tag	codelistType	Type of codelist	1	CharacterString	Must be one of: open enumeration open dictionary closed dictionary
Tag	URI	Identifies the dictionary for open or closed dictionary codelists	0..1	CharacterString	Only for open or closed dictionary codelists
Tag	encoding	Encoding hint for extra values	0..1	CharacterString	Only for open enumeration or open dictionary codelists

3-5.3.12 S100_GF_EnumerationType

S100_GF_EnumerationType and S100_GF_EnumerantClass together model the enumerations defining the allowed values for an enumeration attribute and their semantics. An instance of an enumeration type may define the set of allowed values for an enumeration or codelist attribute, or both.

3-5.4 Relationships between named types

3-5.4.1 Introduction

This subclause describes relationships between object types in more detail. Relationships are classified as follows:

1) Generalisation / Specialisation of feature types and information types.
2) Associations between feature types and information types.

3-5.4.2 GF_InheritanceRelation

The class GF_InheritanceRelation is not realized in the S-100 GFM but object inheritance is allowed through the use of an identical association on the class S100_GF_FeatureType and the class S100_GF_InformationType (see Figure 3-3). The multiplicity of the superType end of the association is such that a subtype may have only one supertype. This is to prevent the modelling of multiple inheritance. The inheritance relation association is modelled at the level of the concrete class rather than on the abstract class S100_GF_NamedType. This prevents a feature type inheriting from an information type and vice versa.

Inheritance associations exist only between named types (classes) and not between named type instances (that is entities occurring in a dataset).

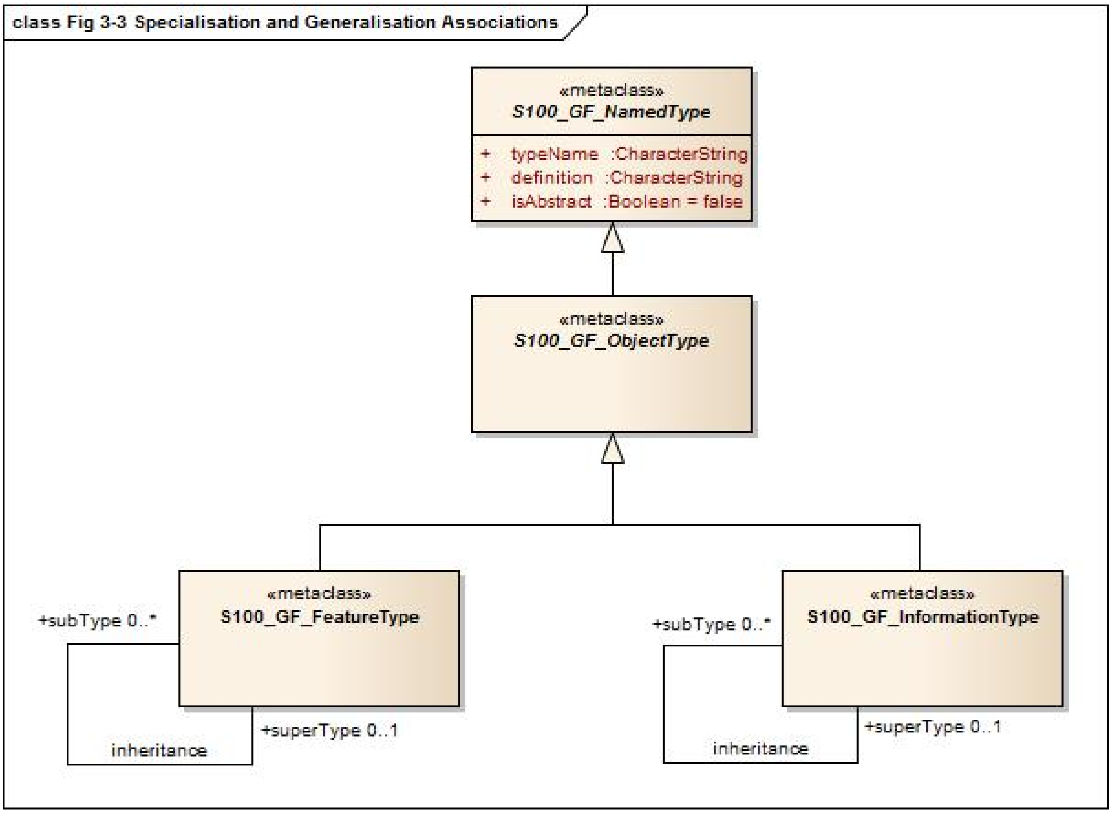


Figure 3-3 — Specialisation and Generalisation Associations

3-5.4.3 S100_GF_AssociationType

Associations are defined by the class S100_GF_AssociationType with two roles and a definition. The ISO 19109 classes GF_AggregationType, GF_SpatialAssociationType, and GF_TemporalAssociationType are not realized explicitly in the S-100 GFM. These classes can be used only if an association is allowed to carry properties. The ISO 19109 GFM allows this because GF_AssociationType is a sub-type of GF_FeatureType. However, S100_GF_AssociationType is not a sub-type of S100_GF_FeatureType.

3-5.4.4 Associations to information types

An association between S100_GF_ObjectType and S100_GF_InformationType is introduced in the S-100 GFM. The role additionalInformation is the default for this association in the S-100 GFM and means that additional information is available for a named type.

3-5.4.5 Default names for association ends

Application Schemas may specify names for association ends (role names). If names are not explicitly provided, the following defaults shall be used.

1) If only one end of an association is given an explicit name “<rolename>”, the other end shall have the default name “inv_<rolename>”.
2) If neither end of the association is given an explicit name, the default role name is “the<target class name>” in which the target class is referenced from the source class.
3) The above rules may not result in a distinct name for each association end in an Application Schema, so Product Specifications may define different or additional rules if needed.
4) If standard names are desired, the following defaults may be used instead of those listed above.
    a. The role “additionalInformation” is a default role name for associations from feature to information types.
    b. Feature/feature or information/information associations navigable in only one direction may use the default end names “source” and “target”. The name “associatedWith” may be used at both ends of a bidirectional association.

Product Specifications may mix individual and standard defaults but must be unambiguous about which name applies to any particular association end.

3-5.5 Behaviour of feature types

The behaviour of feature types is described by operations that may be performed upon or by instances of a feature type. Operations apply only to the interoperability model and do not apply to the data transfer model.

3-5.6 Constraints

Constraints may be introduced to ensure the integrity of the data. Constraints restrict the freedom in an application to prevent creation of erroneous data by specifying combinations of data that are either allowable or not allowable. An Application Schema shall identify constraints in an unambiguous manner.

Only named types and properties may have constraints.

3-6 Rules for Application Schema (ISO 19109 Clause 8)

3-6.1 The application modelling process (ISO 19109 Clause 8.1)

The Application Schema serves two purposes:

1) It achieves a common and correct understanding of the content and structure of data within a particular application field.
2) Secondly, it may provide a computer readable Schema for applying automated mechanisms for data management.

The two roles imply a stepwise process for creating an Application Schema. The steps can be briefly described as:

1) Surveying the requirements from the intended field of application (Universe of Discourse)
2) Making a conceptual model of the application with concepts defined in the GFM. This task consists of identifying feature types, their properties and constraints.
3) Describing elements of the Application Schema in a formal modelling language where necessary. S-100 Application Schemas shall be described using the UML according to rules defined in this part of S-100.
4) Integrating the formal Application Schema with other standardized Schemas, (Spatial Schema, Quality Schema, etc.) into a complete Application Schema.

3-6.2 The Application Schema (ISO 19109 Clause 8.2)

3-6.2.1 Conceptual Schema language for Application Schemas

If a conceptual language is used to design a S-100 Application Schema, then this must be UML.

3-6.2.2 Main rules

The data structures of the Application Schema shall be modelled in the Application Schema.

All classes used within an Application Schema for data transfer shall be instantiable. This implies that the integrated class must not be stereotyped <<interface>>.

3-6.2.3 Identification of Application Schemas

1) The identification of each Application Schema shall include a name and a version. The inclusion of a version ensures that a supplier and a user agree on which version of the

Application Schema describes the contents of a particular dataset. A system of defining unique names and versions for S-100 Application Schemas shall be defined.

2) In UML, an Application Schema shall be described within a PACKAGE, which shall carry the name of the Application Schema and the version stated in the documentation of the PACKAGE.

3-6.2.4 Documentation of an Application Schema

1) An Application Schema shall be documented. A means of documenting Application Schemas for S-100 shall be defined in order to ensure consistency across S-100 Product Specifications.
2) The documentation of an Application Schema in UML may utilise the documentation facilities in the software tool that is used to create the Application Schema, if this information can be exported.
3) If a CLASS or other UML component corresponds to information in a Feature Catalogue, the reference to the Catalogue shall be documented.
4) Documentation of feature types in an Application Schema shall be in a Catalogue with a structure derived from the GFM, such as in a Catalogue in accordance with S-100 Part 5. This could be in text format or XML accompanied by a style sheet (XSLT) used to create a text version.

3-6.3 Rules for Application Schema in UML (ISO 19109 Clause 8.3)

3-6.3.1 Main rules (ISO 19109 Clause 8.3.1)

The main rules for Application Schemas in UML are:

1) An instance of S100_GF_NamedType shall be implemented as a CLASS.
2) An instance of S100_GF_ObjectType shall be implemented as a CLASS.
3) An instance of S100_GF_FeatureType shall be implemented as a CLASS.
4) An instance of S100_GF_InformationType shall be implemented as a CLASS
5) An instance of S100_GF_FeatureAssociationType has the role of linkBetween in association to instances of S100_GF_FeatureType being implemented as CLASSes. It shall be implemented as one of the following cases:
   a) Case 1: An instance of S100_FeatureAssociationType that is not associated with any instances of S100_GF_ThematicAttributeType shall be implemented as an ASSOCIATION between these CLASSes.
   b) Case 2: An instance of S100_FeatureAssociationType that is associated with one or more instances of S100_GF_ThematicAttributeType shall be implemented as an ASSOCIATION CLASS; the associated instances of S100_GF_ThematicAttributeType shall be implemented as ATTRIBUTES of the ASSOCIATION CLASS.
6) An instance of S100_GF_InformationAssociationType has the role of informationLink in association to instances of S100_GF_FeatureType or S100_GF_InformationType being implemented as CLASSES. It shall be implemented one of the following cases:
   a) Case 1: An instance of S100_InformationAssociationType that is not associated with any instances of S100_GF_ThematicAttributeType shall be implemented as an ASSOCIATION between these CLASSes.
   b) Case 2: An instance of S100_InformationAssociationType that is associated with one or more instances of S100_GF_ThematicAttributeType shall be implemented as an ASSOCIATION CLASS; the associated instances of S100_GF_ThematicAttributeType shall be implemented as ATTRIBUTES of the ASSOCIATION CLASS.
7) An instance of S100_GF_AttributeType shall be implemented as an ATTRIBUTE.
8) An instance of S100_GF_SimpleAttributeType shall be implemented as an ATTRIBUTE.

9) An instance of S100_GF_ComplexAttributeType shall be implemented as a CLASS. The instantiated CLASS shall have one or more instances of S100_GF_SimpleAttributeType and/or S-100_GF_ComplexAttributeType as its ATTRIBUTE(s).
10) An instance of the association inheritanceRelation shall be represented by a UML GENERALISATION relationship.

3-6.4 Domain profiles of standard Schemas in UML (ISO 19109 Clause 8.4)

3-6.4.1 Rules for adding information to a standard Schema

Standard Schemas shall not be extended within Application Schemas. Standard Schemas are those that are documented in S-100, for example the Spatial Schema, Feature Catalogue Schema etc.

3-6.4.2 Restricted use of standard Schemas

For some standard Schemas, for example S-100 Part 7 (Spatial Schema), it is possible to redefine the Schema in such a way that only selected parts of the Schema will be used, and only some of the definitions of classes and relationships will be used.

1) Specification of a restricted profile of a standard Schema shall be described in a new UML package by copying the actual definitions (classes and relationships) from the standard Schema. Attributes and operations within classes may be omitted.
2) Reduction of a standard Schema shall be in accordance of the conformance clause given for the actual standard.

3-6.4.3 Rules for use of metadata Schema (ISO 19109 Clause 8.5)

The metadata Schema defined in S100 Part 4 is an application Schema for metadata data sets. Metadata are data describing and documenting data. Metadata for geographic data typically provides information about their identification, extent, quality, spatial and temporal aspects, spatial reference and distribution.

Metadata types shall be implemented as complex attributes that realize elements from S100 Part 4. Thus metadata attributes shall be thematic attribute types.

3-6.4.4 Temporal rules (ISO 19109 Clause 8.6)

S-100 does not include a profile of ISO 19108. Temporal attributes shall be modelled using the types Date, Time or DateTime, S100_TruncatedDate, or complex attributes using combinations of these temporal types. Use of these types makes the attribute an instance of S100_GF_SimpleAttributeType or S100_GF_ComplexAttributeType, as appropriate.

3-6.5 Spatial rules (ISO 19109 Clause 8.7)

3-6.5.1 General spatial rules (ISO 19109 Clause 8.7.1)

The value domain of spatial attribute types shall be in accordance with the specifications given by S-100 Part 7, which provides conceptual Schemas for describing the spatial characteristics of features and a set of spatial operators consistent with these Schemas.

S-100 Part 7 explicitly excludes topological primitives and consequently any topology rules set out in clause 8.7 of ISO 19109 are not relevant in this profile.

3-6.5.2 Spatial attributes

1) Spatial characteristics of a feature shall be described by one or more spatial attributes. In an Application Schema, a spatial attribute is a subtype of a feature attribute (see 5.3), and the taxonomy of its values is defined in the S-100 Part 7.
2) A spatial attribute shall be represented in an Application Schema in either of two ways:
    a) Case 1: as an ATTRIBUTE of a UML CLASS that represents a feature, in which case the ATTRIBUTE shall take one of the spatial objects defined in the Spatial Schema, ISO 19107, as the data type for its value; or

b) Case 2: as a UML ASSOCIATION between the class that represents a feature and one of the spatial objects defined in the Spatial Schema, ISO 19107.

3) A spatial attribute shall take a spatial object as its value. Spatial objects are classified as geometric objects, which are sub-classed as primitives, complexes or aggregates (for geometric objects). The value types of spatial attributes must be the types described in Part 7, or their subtypes.

3-6.5.3 Spatial Quality

The positional quality of a spatial object shall be described by a one way association to a S100_GF_InformationType which is associated with a S100_GF_ThematicAttributeType carrying positional accuracy.

3-6.5.4 Geometric aggregates and complexes to represent spatial attributes of features

3-6.5.4.1 Introduction

The spatial configuration of many features cannot be represented by a single geometric primitive. The types GM_Aggregate and GM_Complex support the representation of such features as collections of geometric objects.

3-6.5.4.2 Geometric aggregates

The spatial profile of S-100 only supports the GM_Multipoint geometric aggregate type. GM_Multipoint shall be used as the value of a spatial attribute that represents a feature as a set of points.

3-6.5.4.3 Geometric complexes

Geometric complexes are used to represent the spatial characteristics of a feature as a set of connected geometric primitives. In addition, instances of GM_Complex allow geometric primitives to be shared by the spatial attributes of different features. There are no explicit links between the GM_Primitives in a GM_Complex; the connectivity between the GM_Primitives can be derived from the coordinate data.

1) A GM_Complex shall be used as the value for a spatial attribute that represents a feature as a collection of connected GM_Objects, which are disjoint except at their boundaries. Subclasses of GM_Complex may be specified to constrain the structure of the GM_Complex used to represent a particular spatial configuration.

2) Features that share elements of their geometry shall be represented as GM_Complexes that are subcomplexes within a larger GM_Complex.

3-6.5.4.4 Geometric composites

A geometric composite is a geometric complex that has all the properties of a geometric primitive except that it is composed of smaller geometric primitives of the same kind. Geometric composites are used to represent complex features that are composed of smaller geometric objects that have the same kind of geometry. A GM_Composite shall be used to represent a complex feature that has the geometric properties of a geometric primitive.

3-6.5.4.5 Features sharing geometry

Different features can share, partly or completely, the same geometry when they appear to occupy the same position. To share a common geometry, spatial feature attributes must share one or more GM_Objects.

There are two ways to share geometry. Complete sharing occurs when two feature instances both take the same instance of a GM_Object as the value of a spatial attribute. This can be required, or precluded, by stating a constraint in the Application Schema. In the absence of such constraints, it may be done whenever necessary.

1) An Application Schema may require instances of two or more feature types to share their geometry completely by including a constraint that the GM_Objects representing the features must be equal.

2) An Application Schema may preclude instances of two or more feature types from sharing their geometry completely by including a constraint that the GM_Objects representing the features are not equal.

3-6.6 Cataloguing rules (ISO 19109 Clause 8.8)

3-6.6.1 Introduction (ISO 19109 Clause 8.8.1)

A Feature Catalogue is a repository that describes real world phenomena of significance to a particular domain. A feature cataloguing methodology provides the details about the organisation of the data that represents these phenomena in categories so that the resulting information is as unambiguous, comprehensible and useful as possible.

3-6.6.2 Application Schema based on a Feature Catalogue (ISO 19109 Clause 8.8.2)

An S-100 Application Schema shall be completely constructed by the definitions provided by a Feature Catalogue implementing the S-100 Feature Catalogue profile.

3-6.6.3 Character encoding

The character encoding used in a dataset shall be defined in the Application Schema. Where more than one character encoding is used the Application Schema shall document how they are used in the dataset.

3-6.7 Codelists

Application Schemas which use an attribute of codelist type shall include a CLASS with tags as specified in Table 3-17 below. The codelist types are described in Part 1.

Table 3-17 — Tags for codelist types

Codelist type	Tags and values
open enumeration	codelistType=open enumeration encoding=other: [something]
closed dictionary	codelistType=closed dictionary URI=<dictionary URL>
open dictionary	codelistType=open dictionary URI=<dictionary URL> encoding=other: [something]

The normative form of the “other: [something]” encoding shall be a character string in the format specified below:

The word ‘other’ followed by a colon and a single space character (that is ‘other: ’ without quotes), followed by one or more alphanumeric strings separated by single spaces.

The normative pattern specifying the portion following ‘other: ’ is specified as (using XML Schema 1.0/1.1 patterns):

```
[a-zA-Z0-9]+( [a-zA-Z0-9]+)*
```

Note that the left parenthesis is followed by a single space and the pattern ends with the asterisk.

Examples:

Table 3-18 — Examples of “extra” values for codelist attributes

`other: loxodromic`	allowed
`other: Seeschifffahrtsstraßen Ordnung`	not allowed (contains the character ß which is not in the allowed set)
`other: German Shipping Regulations`	allowed

`other: German  Shipping Regulations`	not allowed (2 consecutive spaces)
`German Shipping Regulations`	not allowed (does not begin with “other: “)
`other: 287`	allowed
`other: 1,3,5-Trinitroperhydro-1,3,5-triazine`	not allowed (hyphen and comma characters are not in the allowed set)

3-7 Application Schema for Coverages (informative)

3-7.1 Introduction

This rule set for Application Schemas is aimed at Application Schemas for feature oriented data. However, Application Schemas may also be defined for coverages.

This section includes examples of how Application Schemas may be defined for imagery and gridded data. The components of the Application Schemas are defined in ISO 19123 not ISO 19109. However, a coverage may be based on feature type geometries and, in such cases, is conceptually similar to a feature collection. Such feature oriented coverages are discussed below.

3-7.2 Gridded Data

This Application Schema defines a quadrilateral grid coverage with associated metadata. The metadata is generically referenced to ISO 19115-1 and 19115-2. A specific choice of metadata has not been made in this Schema. This Schema can serve for both "matrix" and "raster" data according to the metadata chosen.

The gridded data consists of a single feature - the "image" or "matrix" together with associated metadata taken from MD_Metadata (or MI_Metadata). The CV_Coverage (that is, its relevant sub-type, for example, CV_ContinuousQuadrilateralGridCoverage) serves as the spatial attribute of the gridded data set. It defines an area that is "covered" by the coverage function. For the continuous coverage defined in this Application Schema, the coverage function returns a value for every point in the area covered based on an interpolation function. The Grid Value Matrix is a set of values which drives the interpolation function. It this case the value matrix is a grid traversed by a linear scan (x,y) traversal rule. The spatial referencing is defined by the coordinate reference system.

This template Application Schema supports the majority of imagery and gridded data applications.

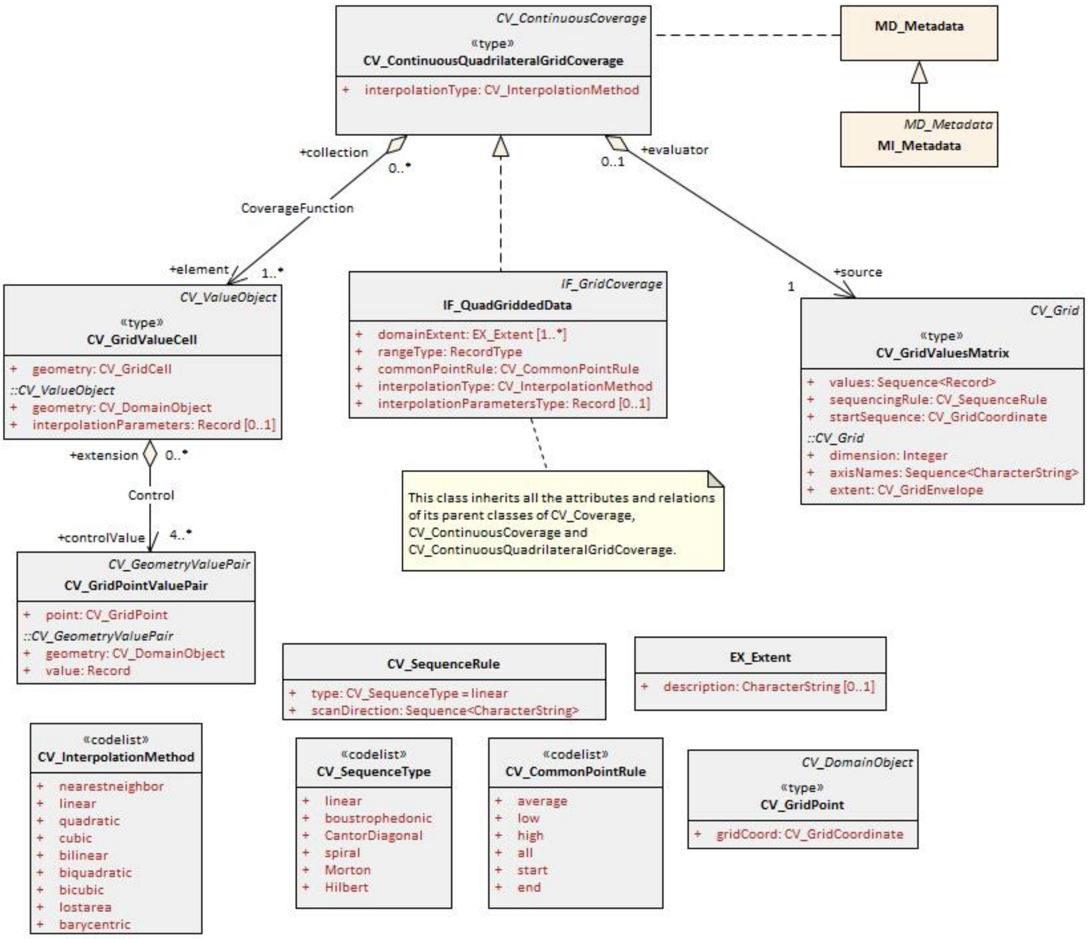


Figure 3-4 – Template Application Schema for a Quadrilateral Grid Coverage

3-7.3 Variable Cell Size Grid

This Application Schema describes a grid of variable cell size (ISO 19123). The traversal order is the Morton order in order to permit support of three (or more) dimensions. This is of particular use for hydrographic data where large volumes of sonar data result in an extensive bottom cover in a 3D grid, but where the cells of similar depth can easily be aggregated.

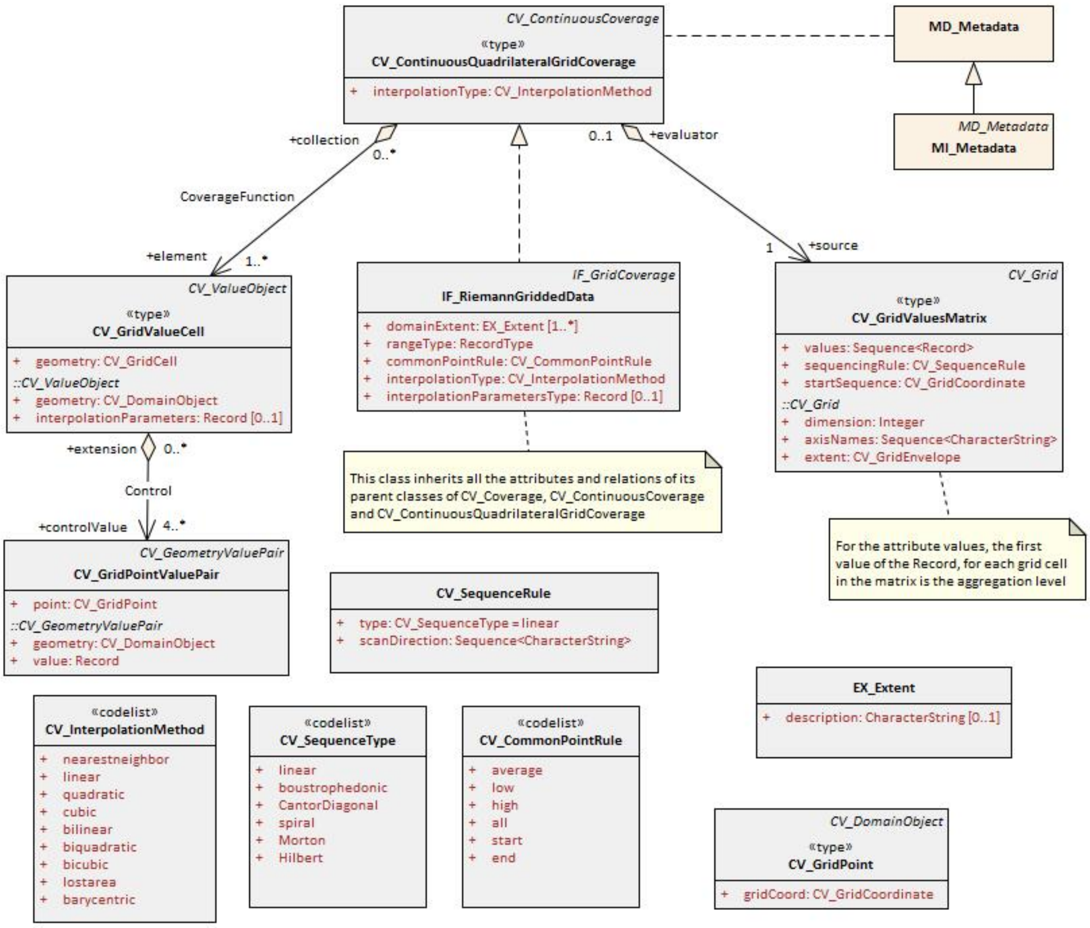


Figure 3-5 – Template Application Schema for a Reimann Grid Coverage

3-7.4 Feature Oriented Image

All gridded data sets are feature oriented, in that a coverage is a subtype of a feature. This means that an entire gridded data set can be considered to be a single feature. A feature structure can be applied to gridded data in two different ways. First, a discrete coverage can carry a feature code as an attribute. For example, a coverage corresponding to the postal code system will have discrete values for each postal code, yet still cover the country completely. The only difference in the Application Schema is a relationship between the discrete coverage and the feature.

The template Application Schema in Figure 3-6 below depicts both the “discrete point” and “discrete grid point” coverage classes. The typical Product Specification would choose one or the other (or both) depending on the type of coverage needed.

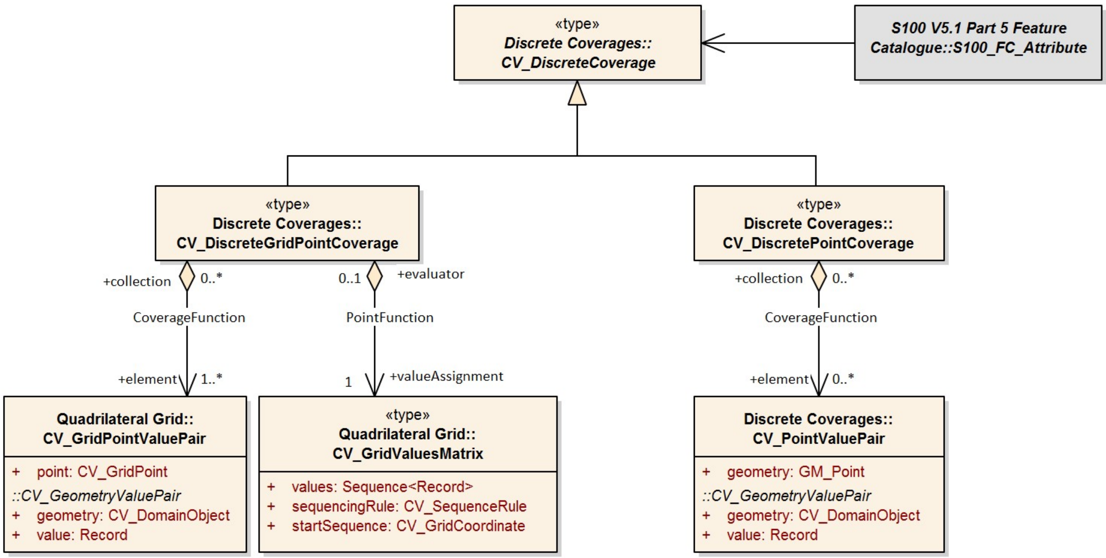


Figure 3-6 – Feature oriented discrete coverage

The second method of establishing a feature structure is to develop a composite data set that contains many separate but adjoining coverages. The coverages may be continuous or discrete. This is very much like the way a "vector" data set is composed where each feature has its own geometry and attributes. In fact vector data may be mixed with coverage data in the same data set. The Application Schema simply allows multiple instances of feature.

Geometric elements such as grids may be shared between multiple features, and features may be related by composition or other relationships as allowed in the general feature model of ISO 19109. A complex feature may include both a continuous grid coverage and vector data such as a polygonal boundary. A feature oriented data set may contain both a continuous coverage of the ocean as collected by sonar, and point and line features corresponding to navigational aids. Topological primitives may relate all of the features. This allows for some interesting and useful structures. A Raster Nautical Chart may include additional vector data describing the navigational aids, hazards and danger zones, which are not "visible" in that they are not portrayed, but which are active in the use of the Raster Nautical Chart, so the mariner can determine whether a ship is within a danger zone, or perform other ECDIS functions.

3-8 S-100 Temporal Framework

3-8.1 Temporal definitions

S-100 does not include a complete profile of ISO 19108. Temporal attributes shall be modelled using the primitive types Date; Time or DateTime; S100_TruncatedDate; S100_IndeterminateDate; or complex attributes using combinations of these temporal types. Use of these types makes the attribute an instance of S100_GF_SimpleAttributeType or a component of an instance of S100_GF_ComplexAttributeType, as appropriate.

Time instants are usually defined either as UTC or as local time with standardized offset to UTC or time zone, using the Gregorian calendar. Therefore S-100 does not implement any of the ISO 19108 Section 5.3 Temporal Reference Systems provisions.

- An S-100 “instant”, in the ISO 19108 sense (ISO 19108 5.2.3.2), is implemented as a single S-100 Date, Time or DateTime (see S-100 Part 1, clause 1-4.5.3.10 S100_TM_Instant).
- An S-100 “period” (TM_Period ISO 19108 5.2.3.3) is implemented as a continuous interval between two S100_TM_Instants (see S-100 Part 1, clause 1-4.5.3.11), expressed as an appropriate complex attribute, for example *fixedDateRange*.
- S100_IndeterminateDate is an instant which lies within a defined period.

- An S100_TM_Instant specified using TruncatedDate can represent an infinite number of ISO 19108 instants.

3-8.2 Temporal relationships

S-100 does not implement full provision of ISO 19108's temporal topology, nor temporal relative positions; however relationships between two S-100 instant and/or period values A and B shall be defined and described using the ISO 19108 TM_RelativePosition values as per Table 3-19 below. Only entries marked with (†) may be meaningfully defined when the S100_TM_Instant is an S100_IndeterminateDate expressed in indeterminate format with a bounded beginning or end.

Table 3-19 — Relationships between temporal values (from from ISO 19108 5.2.3.5 TM_Relative Position)

A~B	A (Single S100_TM_Instant)	A (Single S100_TM_Period)
B (Single S100_TM_Instant)	Before ($A < B$) Equals ($A = B$) After ($A > B$)	Before ($A_{end} < B$)† EndedBy ($A_{end}=B$) Contains ($A_{begin}<B$ and $A_{end}>B$) BegunBy ($A_{begin} = B$) After ($A_{begin} > B$)†
B (Single S100_TM_Period)	Before ($A<B_{start}$)† Ends ($A=B_{end}$) Within ($A>B_{begin}$ and $A<B_{end}$) Begins ($B_{begin} = A$) After ($A > B_{end}$)†	Before ($A_{end} < B_{begin}$)† Meets ($A_{end} = B_{begin}$) Overlaps ($A_{begin}<B_{begin}$ and $A_{end} > B_{begin}$ and $A_{end} < B_{end}$) Begins ($A_{begin} = B_{begin}$ and $A_{end} < B_{end}$) BegunBy ($A_{begin}=B_{begin}$ and $A>B_{end}$) During ($A_{begin} > B_{begin}$ and $A_{end} < B_{end}$) Contains ($A_{begin} < B_{begin}$ and $A_{end} > B_{end}$) Equals ($A_{begin}=B_{begin}$ and $A_{end} = B_{end}$)† OverlappedBy ($A_{begin} > B_{begin}$ and $A_{begin} < B_{end}$ and $A_{end} > B_{end}$) Ends ($A_{begin} > B_{begin}$ and $A_{end} = B_{end}$) EndedBy ($A_{begin} < B_{begin}$ and $A_{end} = B_{end}$) MetBy ($A_{begin} = B_{end}$) After ($A_{begin} > B_{end}$)†

S-100's TruncatedDate enables regular time periods or instants to be defined. These are a union of a, potentially infinite, series of non-intersecting periods or instants. for example:

"2021---05" = { *2021-01-05*, *2021-02-05*, *2021-03-05*, *2021-04-05*, 2021-05-05, *2021-06-05*, *2021-07-05*, *2021-08-05*, *2021-09-05*, *2021-10-05*,*2021-11-05*,*2021-12-05*}

Truncated dates allow a series of periods to be defined. The relationships definable between a single TM_Instant or TM_Period (A) with a set of periods defined by a date in Truncated Format (B) (that is, relationship A~B) are defined in Table 3-20 below (the quantifier "∀" means "for all" and refers to all TM_Periods within B (note some relationships (Before/After) may not be defined when B is a set of unbounded regular time periods).

Table 3-20 — Relationships between temporal values and truncated date

A~B	A (Single TM_Instant)	A (Single TM_Period)
B_n (Multiple TM_Periods)	***Before*** *($A < B_{start}$)* $\forall B$ ***Within*** *($A>=B_{begin}$ and $A<=B_{begin}$)* for some B_i ***After*** *($A > B_{end}$))* $\forall B$	***Before*** *($A_{end} < B_{begin}$)* $\forall B$ ***Within*** *($A_{begin}>=B_{begin}$ and $A_{end}<=B_{end}$)* for some B_i ***After*** *($A_{begin} > B_{end}$)* $\forall B$

3-8.3 Interpretation of models of time intervals and period

The start and end instants of periods (and intervals) shall be included in the period (or interval) unless a Product Specification specifies a different interpretation. This is based on ISO 8601:2004 § 2.1.3 which defines time interval as "the part of the time axis delimited by two instants" and provides that "A time interval comprises all instants between the two limiting instants and, unless otherwise stated, the two limiting instants themselves". Use of "before" or "after" attributes for intervals is not permitted.

The start and end instants are defined by the date/time component of smallest granularity. For example, if the month is the smallest component given in an end instant, the end instant is the whole month and the interval ends at the end of the last day of the month.

Examples: Applying this to encoding intervals using the reduced accuracy representation or the truncatedDate type, results in the interpretations in Table 3-21. The table also indicates how the special case of leap years can be handled.

Table 3-21 — Examples of periods

<S100_TruncatedDateAttributeType> periodStart	`----01--` year and day not encoded	000000 on January 1 through 240000 on the 29th day of February in leap years and the 28th day of February in non-leap years
<S100_TruncatedDateAttributeType> periodEnd	`----02--` year and day not encoded	
<S100_TruncatedDateAttributeType> periodStart	`----0101` year not encoded	000000 on January 1 through 240000 on the 28th day of February each year
<S100_TruncatedDateAttributeType> periodEnd	`----0228` year not encoded	
<S100_DateAttributeType> dateStart	`20120105`	000000 on January 5, 2012 through 240000 on June 18, 2012
<S100_DateAttributeType> dateEnd	`20120618`	

3-9 Use of format-specific types for truncated dates

Data formats may utilise specific types as supported by that format in order to incorporate truncated values. Where this occurs the format description must specify the mapping between the S100_TruncatedDateAttributeType values and those of the format-specific types.

Example: An XML based encoding may use the *gMonthDay* simple attribute type (which is an XML Schema built-in type) as an equivalent representation for "December 17 each year":

xs:gMonthDay: `--12-17`

This is equivalent to the value `----1217` in a data format which adheres strictly to S-100.

3-10 Instance Identifiers

Identifiers of instances should utilize the Maritime Resource Name (MRN) concept and namespace. The MRN namespace is administered by the International Association of Lighthouse Authorities (IALA) through the website http://mrnregistry.org, which also contains references to the full set of rules that apply to the MRN concept. The topmost namespace urn:mrn remains fixed, with subsequent name spaces separated by colons, and available through the application process explained on the website. Any organization wishing to issue MRN conformant identifiers should apply for a name space from IALA, or from an organization that already has a namespace registered.

For example, IHO applies for a namespace, and subsequently gives all member states a sub-namespace under the urn:mrn:iho namespace; for NOAA this could be urn:mrn:iho:us and for CHS this could be urn:mrn:iho:ca. NOAA and CHS would then administer their respective namespaces as needed and within the MRN rules.

The following rules apply to the MRN namespace.

The Namespace Specific String (NSS) of all URNs that use the "mrn" NID shall have the following structure:

```
<URN>   ::= "urn:mrn:" <OID> ":" <OSS>
<OID>   ::= 1*(ALPHA / DIGIT) ; Organizational ID
<OSS>   ::= <OSNID> ":" <OSNS> ; Organizational specific string
<OSNID> ::= 1*(ALPHA / DIGIT / "-") ; Organizational specific namespace ID
<OSNS>  ::= 1*<URN chars> ; Organizational specific namespace string

DIGIT   ::= %x30-39 ; 0-9
ALPHA   ::= %x61-7A ; a-z
```

Basics of the ABNF notation used:

" "	literals (terminal character strings); terms not in quotes are non-terminals
/	alternatives
()	indicates a sequence group, used as a single alternative or as a single repeating group
<a>*<b>	indicates that the following term or group can repeat at least <a> and at most <b> times; default values are 0 and infinity, respectively
;	comment

The entire URN is case-insensitive.

<URN chars> As defined in RFC2141

The process for assigning unique organizational IDs is managed by IALA. Details and application process can be found at <http://www.mrnregistry.org>.

S-100 – Part 4a

Metadata

Page intentionally left blank

Contents

Page intentionally left blank

4a-1 Scope

The S-100 metadata profile described in Parts 4a, 4b and 4c provides a specification for describing, validating and exchanging metadata about geographic datasets commonly produced by hydrographic organizations. Its purpose is the creation of metadata records that provide information about the identification, spatial and temporal extent, quality, Application Schema, spatial reference system, and distribution of digital geographic data. It is applicable to the cataloguing of datasets, clearinghouse activities, and the full description of geographic and non-geographic resources. Although it is primarily intended to describe digital geographic data, it may also be used to describe other resources such as charts, maps, images, textual documents and non-geographic resources. It makes provision for the description of; *attributes*, *attributeTypes*, *features*, *featureTypes*, *collectionHardware*, *collectionSession*, *datasets*, *dataset series*, *nonGeographicDatasets*, *propertyTypes*, *fieldSession*, *software* and *services*. It should be noted that this profile is not limited to the resources listed in the ISO 19115-1 codelist *MD_ScopeCode* <<*Codelist*>> (ISO 19115-1 - B.3.28), and can be extended to include additional resources if required.

This profile is based on ISO 19115-1 Metadata and 19115 Part 2 - Metadata for imagery and gridded data. It also takes account of ISO/TS 19115-3 Metadata – XML Schema implementation for Fundamental Concepts.

ISO 19115-1 provides an abstract structure for describing digital geographic information by defining metadata elements and establishing a common set of metadata terminology, definitions, and extension procedures. ISO/TS 19115-3 provides an eXtensible Markup Language (XML) implementation of ISO 19115-1, and guidance for developing profiles and extensions.

This document is intended for developers and implementers of metadata applications, and provides a basic understanding of the principles and the overall requirements for standardisation of geographic information. It should be used in conjunction with the standards listed under clause 4a-4 – Normative references.

Further information concerning S-100 metadata implementation, encoding and quality principles are included in the following associated documents:

1) S-100 Part 4b – Metadata Extensions for Imagery and gridded data;
2) S-100 Part 4c – Metadata Quality Principles;
3) Appendix 4a–C - Metadata Implementation.

4a-2 Conformance

4a-2.1 Conformance of this Profile with other Standards

In addition to the elements listed in ISO 19115-1, this profile also adopts all associated 19115-1 obligations and conditions, with the exception of the *metadataIdentifier* element which has been changed from optional to mandatory. This has been done to facilitate the implementation and management of metadata records by allowing instances of duplicate metadata records to be identified, and defining the relationship of a child metadata record with its parent metadata record. The specifics of any metadata hierarchy relationships will be detailed in the Product Specifications.

Taking into account the change identified above, and the requirements documented in ISO 19106:2004, this Profile meets the requirements of conformance class 1[1]. The Profile is a community profile[2] of ISO 19115-1 and includes an extension in the context permitted by the base standard[3].

This profile includes *parentMetadata* as a core metadata element for geographic datasets. If a dataset metadata record has a parent metadata record, then this element becomes mandatory and therefore

1 Conformance class 1 as described at Section 2 *Conformance* and Appendix B.3 *Example of a profile with specialisations* (ISO 19106:2004).

2 A profile of a single base standard can include a subset, which is equivalent to the entire base standard. That is, a subset can equal the whole (19106:2005, p15).

3 This conforms to the rules included at Annex C (ISO 19115-1:2014).

should be considered a 'core' element. Guidance on the XML implementation of this profile is included at Appendix 4a-C.

4a-2.2 Backward compatibility

According to ISO 19115-1:2014, ISO continues to make the UML models from ISO 19115:2003/Cor 1:2006 available for use. Backward compatibility is to be provided using a transformation service.

4a-3 Conformance to this Profile

Any metadata claiming conformance to this Profile shall:

1) Have content according to the data dictionary definitions in Annex B of ISO 19115-1, (including changes required by ISO 19115-1 Amendment 1:2018) with the exception of the metadata element *metadataIdentifier* which has a mandatory obligation;
2) Prove conformance by validating XML document instances against the S-100 Metadata Profile Schemas which are available from the IHO website at Profiles based on this Profile.

All product specific implementations of this profile shall provide an Extensible Stylesheet Language (XSL) transform file/resource that can translate the XML document instances into the S-100 Metadata Profile XML format. These resulting XML document instances shall be validated using the ISO/TS 19115-3 XSDs.

4a-4 Normative references

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

4a-4.1 Profile definition

The following documents were the references used to define the S-100 Metadata Profile:

ISO 19115-1:2014, *Geographic information – Metadata – Part 1 - Fundamentals*

ISO 19115-1/Amdt01:2018, *Geographic information – Metadata – Part 1 - Fundamentals* (Amendment 1)

ISO 19115-2:2009, *Geographic information - Metadata - Part 2: Extensions for imagery and gridded data*

ISO 19119:2016, *Geographic information – Services*

ISO/TS 19115-3:2016, *Geographic information - Metadata - XML schema implementation for fundamental concepts*

4a-4.2 Informative references

ISO 19115:2003, *Geographic information – Metadata*

ISO 19115:2003/Cor.1:2006, *Geographic information - Metadata* (Technical Corrigendum 1)

ISO/TS 19139:2007, *Geographic information - Metadata - XML schema implementation*

4a-5 Requirements

4a-5.1 Business purpose and Intended use

Metadata can satisfy a number of uses:

1) Data Discovery - summary descriptions of content and quality, contact details, off-line distribution and on-line references (URL) for on-line viewing.

2) Data use - more extensive information on data coverage, maintenance, content and details of data creation. It includes additional contact, distribution and quality details.

3) Data Fitness – additional detail about use, limitations, format, age, and extents. This level of metadata assists the user to determine the data's suitability for use.

4) Data Sharing – further detail relating to data content, transfer formats, and spatial representation.

5) Data Management – the most detailed level of metadata, which includes information on the data quality regimes and data quality test results. This type of information is sometimes important when data is exchanged between organizations.

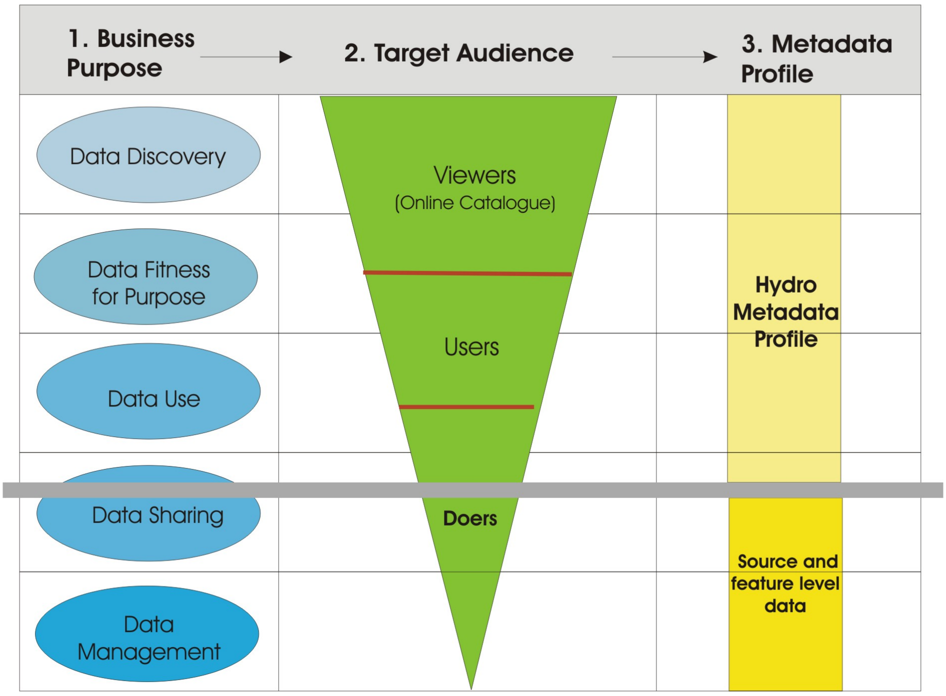


Figure 4a-1— Business Purpose

Figure 4a-1 above illustrates the relationship between the types of metadata required by different user communities, and the scope of this profile. Each S-100 based Product Specification will describe the source and feature-level metadata that will be required to support data use, data sharing, and data management. The more demanding requirements for comprehensive metadata (as illustrated by "Doers" in Figure 4a-1), require further attribution to allow source selection and feature analysis.

ISO 19115-1 does not provide all the metadata necessary to describe imagery. This has been included in Part 2 to ISO 19115, which incorporates elements that are needed for the description of imagery and gridded data. ISO 19130 – "Sensor and data model for imagery and gridded data", is an important standard associated with ISO 19115 Part 2, as it specifies the information required to support the geolocation of georeferenceable imagery, including a sensor description and associated physical information defined by a sensor model, fitting functions, and ground control points. It describes how the sensor measurements and the geolocation information are logically associated. In particular, ISO 19130 describes the sensor and data model for hydrographic sonar requirements, and the associated metadata. This will be described in relevant Product Specifications.

An XML implementation of the ISO 19115-1 which describes how the abstract UML models in ISO 19115-1 are converted into XML is documented in ISO publication ISO/TS 19115-3.

Although this profile is largely based on the above mentioned standards, reference to additional standards will need to be made. (See sections – "Normative References").

This Profile defines:

1) Mandatory and conditional metadata sections, metadata entities, and metadata elements;
2) The minimum set of metadata elements for any resource in order to conform to this Profile;
3) The core metadata for geographic datasets;
4) Optional metadata elements that allow for a more extensive standard description of resources; and
5) The option to extend the Profile to cater for specialised needs.

Implementation of the Profile is based on ISO/TS 19115-3:2016, and includes:

1) The use of the ISO/TS 19115-3:2016 XSDs;
2) XML documents containing dictionaries to implement the ISO 19115-1:2014 codelists (XML data dictionaries of the ISO 19115-1:2014 codelists in GML format); and
3) XML data dictionaries of the S-100 Geographic Extent identifier codelists[4].

While the UML class *S100_Metadata* specialises the class *MD_Metadata*, the specialisation only involves restrictions of the parent class. Hence, for the purpose of XML implementation, the *MD_Metadata* element shall be used to support interoperability with other ISO 19100 standards for geographic information.

4a-5.2 Metadata for describing geographic data and other resources

The Profile identifies the metadata required to describe digital geographic data and resources, and is applicable to independent datasets, dataset aggregations, geographic features, feature classes and attributes. Metadata is documented via the creation of XML document instances, which are validated against the S-100 Metadata Profile XSDs, and relevant codelists and enumerations[5].

If a Product Specification extends the metadata of this profile, the rules in Appendix 4a-E must be followed, and the Product Specification must provide a metadata Schema to validate metadata against.

Metadata records must contain a minimum set of core elements (see Section 4a-5.3 which are necessary for conformance with this Profile. A number of additional elements required for discovery purposes have also been identified and are described in the Appendix 4a-C.

Quality information is important for assessing whether datasets or resources are fit for use, and quality metadata have therefore been documented in Part 4c.

4a-5.3 Obligations/conditions

Obligation descriptors have been included to provide an indication of whether a metadata entity or element must be documented or may be conditionally or unconditionally left to the discretion of the metadata encoder. This descriptor may have the following values: M (mandatory), C (conditional) or O (optional). The following definitions form section B.1.4 *Obligation/Condition* of ISO 19115-1 are included below.

A **mandatory (M)** obligation means the metadata class or metadata element shall be documented.

A **conditional (C)** obligation specifies an electronically manageable condition under which at least one metadata class or a metadata element is mandatory. 'Conditional' is used for one of the three following possibilities:

1) Expressing a choice between two or more options. At least one option is mandatory and must be documented.

[4] Reference to codelists of geographic identifiers to be provided. They do not appear in the ISO 19115 codelists.

[5] Enumeration: A fixed list of valid identifiers of named literal values. Attributes of an enumerated type may only take values from this list (source: ISO 19136:__, *Geographic information — Geography Markup Language (GML)*).

2) Documenting a metadata class or a metadata element if another element has been documented.

3) Documenting a metadata element if a specific value for another metadata element has been documented.

If the answer to the condition is positive, then the metadata entity or the metadata element shall be mandatory.

An **optional (O)** obligation means that the metadata class or the metadata element may be documented or may not be documented. Optional metadata classes and optional metadata elements have been defined to provide a guide to those looking to fully document their data. (Use of this common set of defined elements will help promote interoperability among geographic data users and producers world-wide.) If an optional class is not used, the elements contained within that class (including mandatory elements) will also not be used. Optional classes may have mandatory elements; those elements only become mandatory if the optional entity is used.

4a-5.4 Minimum metadata requirements

The minimum requirements for recording metadata include a number of elements that must be completed in order to conform to this Profile. It should be noted that the obligation is not mandatory for all elements, however some conditional elements may become mandatory under certain conditions (for example *resourceType*).

Table 4a-1 identifies the minimum set of metadata elements that should be completed for datasets and other resources. These elements also form part of the minimum metadata for geographic datasets listed in Table 4a-2.

Page intentionally left blank

Table 4a-1 — Minimum metadata for geographic datasets and other resources

Name	Path	Datasets	Other resources
Metadata file identifier	MD_Metadata.metadataIdentifier > MD_Identifier.code	**M**	**M**
Metadata language	MD_Metadata.defaultLocale > PT_Locale.language	**C** *(documented if not defined by the encoding process)*	**C** *(same as for dataset)*
Metadata character set	MD_Metadata.defaultLocale > PT_Locale.characterEncoding	**C** *(documented if ISO 10646-1, is not used and not defined by the encoding process)*	**C** *(same as for dataset)*
Metadata file parent identifier	MD_Metadata.parentMetadata > CI_Citation.identifier	**C** *(documented if the hierarchy of a higher level exists)*	**C** *(same as for dataset)*
Party responsible for the metadata information	MD_Metadata.contact > CI_Responsibility.CI_Individual (table 4a-2) or MD_Metadata.contact > CI_Responsibility.CI_Organisation (table 4a-3)	**M** *(either organization or individual must be documented)*	**M** *(same as for dataset)*
Date(s) associated with the metadata	MD_Metadata.dateInfo > CI_Date	**M** *(creation date required, other dates may be provided)*	**M** *(same as for dataset)*
Resource title	MD_Metadata.identificationInfo > MD_DataIdentification.citation >CI_Citation.title	**M**	**M** (See note 2)
Resource reference date	MD_Metadata.identificationInfo > MD_DataIdentification.citation > CI_Citation.date > CI_Date.date	**M**	**M** (See note 2)
Resource reference date type	MD_Metadata.identificationInfo > MD_DataIdentification.citation > CI_Citation.date > CI_Date.dateType > CI_DateTypeCode	**M**	**M** (See note 2)
Abstract describing the resource	MD_Metadata.identificationInfo > MD_DataIdentification.abstract	**M**	**M** (See note 2)

Name	Path	Datasets	Other resources
Resource default language	MD_Metadata.identificationInfo > MD_DataIdentification.defaultLocale > PT_Locale.language	**M**	**C** *(only used if MD_DataIdentification has been used)*
Resource default character set	MD_Metadata.identificationInfo > MD_DataIdentification.defaultLocale > PT_Local.characterEncoding	**C** *(documented if UTF-8 is not used)*	**C** *(documented if UTF-8 is not used)*
Topic category	MD_Metadata.identificationInfo > MD_DataIdentification.topicCategory	**M**	**C** *(if resourceType = 'series' topicCategory is mandatory)*
Geographic location of the resource (by description)	MD_Metadata.identificationInfo > MD_DataIdentification.extent > EX_Extent > EX_GeographicDescription.geographicIdentifier > MD_Identifier.code	**C** (See notes 3 and 4)	**O** (See note 4)
West longitude	MD_Metadata.identificationInfo > MD_DataIdentification.extent > EX_Extent > EX_GeographicBoundingBox.westBoundLongitude	**C** (See notes 3 and 4)	**O** (See note 4)
East longitude	MD_Metadata.identificationInfo > MD_DataIdentification.extent > EX_Extent > EX_GeographicBoundingBox.eastBoundLongitude	**C** (See notes 3 and 4)	**O** (See note 4)
South latitude	MD_Metadata.identificationInfo > MD_DataIdentification.extent > EX_Extent > EX_GeographicBoundingBox.southBoundLatitude	**C** (See notes 3 and 4)	**O** (See note 4)
North latitude	MD_Metadata.identificationInfo > MD_DataIdentification.extent > EX_Extent > EX_GeographicBoundingBox.northBoundLatitude	**C** (See notes 3 and 4)	**O** (See note 4)
Name of the scope/type of resource for which the metadata is provided	MD_Metadata.metadataScope > MD_MetadataScope.resourceScope > MD_ScopeCode (codelist – ISO 19115-1)	**M** *(default = "dataset")*	**M**
Description of scope of resource for which the metadata is provided	MD_Metadata.metadataScope > MD_MetadataScope.name	**O**	**O**

NOTE1 ISO 10646-1 - Information technology — Universal Multiple-Octet Coded Character Set (UCS).

NOTE2 MD_ServiceIdentification may be used instead of MD_DataIdentification if hierarchyLevel = ‘service’.

NOTE3 For a geographic dataset, include metadata for the geographic bounding box (West longitude, East longitude, South latitude and North latitude) or the geographic description identifier (The use of geographic bounding box is recommended - see Section 5.6.3).

NOTE4 If any one of west longitude, east longitude, south latitude or north latitude exists, then the remaining three must also be completed.

Table 4a-2 — Individuals

Name	Path	Datasets	Other resources
Name of the individual	CI_Individual.name	**C** *(documented if ‘positionName’* and *‘partyIdentifier’ not documented)*	**C** *(same as for dataset)*
Position of the individual in an organization	CI_Individual.positionName	**C** *(documented if ‘name’ and ‘partyIdentifier’ not documented)*	**C** *(same as for dataset)*
Contact information for the individual	CI_Individual > contactInfo > CI_Contact	**M** (see note 6)	**M** (see note 6)
Identifier for the party	CI_Individual.partyIdentifier	**C** *(documented if ‘name’ and ‘positionName’ not documented*	**C** *(same as for dataset)*

Table 4a-3 — Organisations

Name	Path	Datasets	Other resources
Name of the organisation	CI_Organisation.name	**C** *(documented if 'positionName' not documented* – see Note 5*)*	**C** *(same as for dataset)*
Position of an individual in the organisation	CI_Organisation.positionName	**C** *(documented if 'name' not documented* – see Note 5*)*	**C** *(same as for dataset)*
Contact information for the organisation	CI_Organisation.contactInfo > CI_Contact	**M** (see note 6)	**M** (see note 6)
An individual in the named organisation	CI_Organisation.individual > CI_Individual	**M**	**M**
Identifier for the party	CI_Organisation.partyIdentifier	**C** *(documented if 'name' and 'positionName' not documented*	**C** *(same as for dataset)*

NOTE 5 S-100 restricts ISO 19115-1 in that documenting the 'logo' attribute of CI_Organisation is not sufficient to allow omission of both 'name' and 'positionName'.

NOTE 6 At least one of CI_Contact attributes phone / address / onlineResource / contactInstructions must be documented.

4a-5.5 S-100 compliancy categories

When implementing S-100 support, different systems may have different requirements to S-100 based products and their adherence to the S-100 framework. ECDIS may require a very high degree of compliance; while a reporting system may require a lower degree of compliance by, for example, not needing an S-100 based exchange method. To facilitate a means of communicating the intent of a Product Specification and resulting products an S-100 compliance category can be declared. Four categories are defined.

4a-5.5.1 Category 1 - IHO S-100 object model compliant

The Product Specification contains an object model which is available as a Feature Catalogue from the IHO S-100 GI Registry and is compliant with the S-100 spatial model (S-100 Parts 7 and 8).

4a-5.5.2 Category 2 - IHO S-100 compliant with non-standard encoding

The Product Specification adheres to the minimum requirements of S-100 Part 11. The Product Specification specifies which of the S-100 Part 10 encoding methods is used; or it specifies another encoding, including how it maps to the S-100 GFM. Metadata is according to S-100 Part 4, a profile of Part 4 or an extension according to Part 4 rules.

4a-5.5.3 Category 3 - IHO S-100 compliant with standard encoding

As Level 2 with the following restrictions:

- The Product Specification uses only an encoding method defined in S-100 Part 10.

4a-5.5.4 Category 4 - - IHO S-100 and IMO harmonized display compliant

As Level 3 with the following restrictions:

- Metadata is according to S-100 Part 4 or a profile of Part 4 metadata;
- The Product Specification includes a Portrayal Catalogue available from the IHO S-100 GI Registry;
- The Product Specification includes defined methods for the S-100 defined cyber security scheme (at a minimum including digital signature and, if applicable, the method of encryption);
- Test material is embedded into the Product Specification or test material is available in a separate package. The test cases and related material is at a minimum comparable to IHO Publication S-64 for S-52/S-57/S62/S-63);
- The Product Specification uses a CRS from the EPSG Geodetic Parameter Registry. EPSG CRSs which do not comply with the requirements of S-100 Part 6 or the selected encoding method defined in S-100 Part 10 should not be used;
- If appropriate, the Product Specification includes an Alerts and Indications Catalogue available from the IHO S-100 GI Registry; and
- If appropriate, the Product Specification is compliant with the Interoperability Catalogue available from the IHO S-100 GI Registry.

4a-5.6 Recommended metadata for geographic datasets

Although ISO 19115-1 defines an extensive set of metadata elements, only a subset of these are used. It is essential however that a minimum number of metadata elements be maintained for a dataset (as listed in Table 4a-1). When describing geographic datasets however, it is recommended that additional metadata elements (in addition to the minimum requirements for geographic datasets) be used. This set of metadata, which includes the minimum set of metadata and some additional optional elements, is referred to as **recommended metadata**. Table 4a-4 lists the recommended metadata required to describe a *dataset*, typically for catalogue purposes. This list contains metadata answering the following questions:

1) ‘Does a dataset on a specific topic exist (“what”)?’
2) ‘For a specific place (“where”)?’
3) ‘For a specific date or period (“when”)?’

4) 'A point of contact to learn more about or order the dataset ("who")?'

By using the core metadata described below, interoperability will be enhanced, and potential users should be able to understand without ambiguity the characteristics of geographic datasets or resources.

Table 4a-4 — Recommended metadata for geographic datasets

Name	Path	Obligation
Unique identifier for this metadata record	MD_Metadata.metadataIdentifier > MD_Identifier.code	M [a]
Metadata language	MD_Metadata.defaultLocale > PT_Locale.language	C [b]
Metadata character set	MD_Metadata.defaultLocale > PT_Locale.characterEncoding	C [c]
Metadata file parent identifier	MD_Metadata.parentMetadata > CI_Citation.identifier	C [d]
Party responsible for the metadata information	MD_Metadata.contact > CI_Responsibility.CI_Individual or MD_Metadata.contact > CI_Responsibility.CI_Organization	M
Date(s) associated with the metadata	MD_Metadata.dateInfo > CI_Date	M
Metadata standard name	MD_Metadata.metadataStandard > CI_Citation.title	O
Metadata standard version	MD_Metadata.metadataStandardVersion	O
Dataset title	MD_Metadata.identificationInfo > MD_DataIdentification.citation > CI_Citation.title	M
Dataset reference date	MD_Metadata.identificationInfo > MD_DataIdentification.citation > CI_Citation.date	M
Resource identifier	MD_Metadata.identificationInfo > MD_DataIdentification.citation > CI_Citation.identifier > MD_Identifier.code	O
Abstract describing the data	MD_Metadata.identificationInfo > MD_DataIdentification.abstract	M
Resource point of contact	MD_Metadata.identificationInfo > MD_DataIdentification.pointOfContact > CI_Responsibility	O
Spatial representation type	MD_Metadata.identificationInfo > MD_DataIdentification.spatialRepresentationType	O
Spatial resolution of the dataset	MD_Metadata.identificationInfo > MD_DataIdentification.spatialResolution > MD_Resolution.distance or MD_Resolution.equivalentScale or MD_Resolution.vertical or MD_Resolution.angularDistance or MD_Resolution.levelOfDetail	O [e]
Dataset language	MD_Metadata.identificationInfo > MD_DataIdentification.language	M
Dataset character set	MD_Metadata.identificationInfo > MD_DataIdentification.defaultLocale > PT_Locale.characterEncoding	C [f]
Dataset topic category	MD_Metadata.identificationInfo > MD_Identification.topicCategory	M

Name	Path	Obligation
Geographic location of the dataset (by four coordinates or by description)	MD_Metadata.identificationInfo > MD_Identification.extent > EX_Extent > EX_GeographicBoundingBox or EX_GeographicDescription	C [g, h]
Temporal extent information for the dataset	MD_Metadata.identificationInfo > MD_Identification.extent > EX_Extent.temporalElement	O
Vertical extent information for the dataset	MD_Metadata.identificationInfo > MD_DataIdentification.extent > EX_Extent.verticalElement > EX_VerticalExtent	O
Lineage	MD_Metadata.resourceLineage > LI_Lineage	O
Reference system	MD_Metadata.referenceSystemInfo > MD_ReferenceSystem.referenceSystemIdentifier > RS_Identifier	O
Distribution Format	MD_Metadata.distributionInfo > MD_Distribution > MD_Format	O
On-line link to resource	MD_Metadata.distributionInfo > MD_Distribution > MD_DigitalTransferOption.onLine > CI_OnlineResource	O
Constraints on resource access and use	MD_Metadata.identificationInfo > MD_DataIdentification > MD_Constraints.useLimitations and/or MD_LegalConstraints and/or MD_SecurityConstraints	O
Name of the scope/type of resource for which the metadata is provided	MD_Metadata.metadataScope > MD_MetadataScope.resourceScope	C [i]

a) The Profile imposes a mandatory obligation on the metadata element metadataIdentifier.

b) Language: documented if not defined by the encoding process.

c) characterEncoding: Documented if UTF-8 is not used and not defined by the encoding process.

d) Documented if a higher level of hierarchy level exists (for example if the geographic 'dataset' is part of a 'series').

e) Distance is preferred over equivalentScale because the scale will change when presented at different sizes on a screen. distance or equivalentScale must be documented if available.

f) characterSet: Documented if ISO 10646-1 is not used.

g) Include either the geographic bounding box (extents) or the geographic description (It is recommended that geographic bounding box should be used - see Section 5.6.3).

h) If any one of west longitude, east longitude, south latitude or north latitude exists, then the remaining three must also be completed.

i) Name is mandatory if resourceScope not equal to "dataset".

Source: Adapted from Table 3 - *Core metadata for geographic datasets* (ISO 19115:2005).

Mandatory attributes are nillable.

4a-5.7 Variations and preferences

4a-5.7.1 Metadata element metadataIdentifier

The obligation for the metadata element *metadataIdentifier* is 'optional' in ISO 19115-1, however this profile applies a more stringent obligation and defines an extension to make the obligation '**mandatory**'. Each Product Specification will provide rules for creating file identifiers.

For example, this could support linkage between parent and child metadata records. The identifier code of the child's *parentMetadata/CI_Citation.identifier* element is the same as the identifier code of the parent's *metadataIdentifier* element, thus supporting the hierarchical relationship between metadata records.

4a-5.7.2 Metadata element parentMetadata

The metadata element *parentMetadata* (conditional obligation) is included as a recommended metadata element for describing geographic datasets in the profile. Under certain conditions this metadata element is mandatory. For instance, in some cases dataset metadata may be part of a dataset series. In these circumstances *parentMetadata* shall be populated.

The concept of metadata scope allows a dataset to be described in more than one metadata record. A dataset may be part of a collection, and in this instance, the dataset may be described in two metadata records: as a dataset in its own right and as part of a collection. The dataset may also be more discrete. For example, a chart may be described individually and as part of a collection or (chart series). An organization may choose to produce a metadata record for each chart and a metadata record for the collection (chart series). Further information on metadata scope and their implementation is available in Annex D and Annex E of ISO 19115-1.

4a-5.7.3 Geographic extent of the dataset

The ISO 19115-1 condition for spatial extent determines that if the *resourceScope* is 'dataset' then either the *geographic bounding box* or the *geographic description* is mandatory (ISO 19115-1 Table B.3). To make spatial searches more effective, it is recommended that the extent be described as a geographic bounding box in preference to a geographic description. Completing only the geographic description code may not satisfy the needs of spatial searches as an extent could be ambiguous (for example, 'France' could mean the mainland only or it may include all external territories). However, in other circumstances, the geographic descriptions are clearly defined, and can present a more efficient means of description. Therefore, Product Specifications shall specify how geographic extent of a dataset is described.

4a-5.7.4 Data and Date Time information

Dates for both the metadata and the actual data must be provided. In MD_Metadata, there is a date stamp for the metadata. In the citation, provided as part of MD_Identification, there is a production, publication, or revision date for the dataset. These dates are not necessarily the same. In some cases, one set of metadata may be provided for multiple sets of data, which may have been produced, published or revised at different times. The need for an associated date of origin is not restricted to digital or geographic data. Users who derive results from reprocessed data need to know the version of the data they are using.

4a-5.7.5 Metadata extension information

The *S100_Metadata* class specialises the *MD_Metadata* class, restricting the obligation of *metadataIdentifier* from optional to mandatory. Tables 4a-5 and 4a-6 provide relevant information about the extension for *S100_Metadata.* A modified UML diagram is provided at Appendix 4a-A, the modified values for the data dictionary are provided at Appendix 4a-B (Table B-1 - *Modifications to data dictionary ISO 19115*).

Table 4a-5 — Metadata extension for *S100_Metadata*

MD_MetadataExtensionInformation	
MD_ExtendedElementInformation	
name	S100_Metadata
definition	S-100 Metadata Profile of MD_Metadata

obligation	Mandatory	
condition		
dataType	specifiedClass	
maximumOccurrence	1	
domainValue		
parentEntity	MD_Metadata	
rule	New class	
Rationale	Extension of MD_Metadata to include change of obligation to *fileIdentifier*	
Source	organisationName	International Hydrographic Organization
	role	owner
conceptName	the name of the item (in the IHO GI Registry)	
code	language neutral identifier (code in the IHO GI Registry)	

Table 4a-6— Metadata extension for *S100_Metadata*

MD_MetadataExtensionInformation		
MD_MetadataElementInformation		
name	metadataIdentifier	
definition	ISO 19115-1:2014 Table B.2	
obligation	mandatory	
condition		
dataType	Class	
maximumOccurrence	1	
domainValue	MD_Identifier	
parentEntity	S100_Metadata	
rule	Change obligation to mandatory	
Rationale	To ensure a file identifier is always entered	
Source	organisationName	International Hydrographic Organization
	role	owner
conceptName	the name of the item (in the IHO GI Registry)	
code	language neutral identifier (code in the IHO GI Registry)	

4a-5.8 Metadata for services

The elements to be used for discovery of services are listed in Table 4a-7. The elements are similar to those used for datasets except that SV_ServiceIdentification replaces MD_DataIdentification and two conditional elements are added to document the coupling (if any) between the service and a dataset.

This edition of the S-100 profile of metadata for services does not document the operations proffered by services. Accordingly, the profile omits the optional metadata elements and attributes related to operation information that are defined in ISO 19115-1.

Table 4a-7 — Metadata for the discovery of services

Name	**Path**	**Obligation**
Unique identifier for this metadata record	MD_Metadata.metadataIdentifier > MD_Identifier.code	M a

Name	Path	Obligation
Metadata language	MD_Metadata.defaultLocale > PT_Locale.language	C [b]
Metadata character set	MD_Metadata.defaultLocale > PT_Locale.characterEncoding	C [c]
Metadata parent identifier	MD_Metadata.parentMetadata > CI_Citation.identifier	C [d]
Party responsible for the metadata information	MD_Metadata.contact > CI_Responsibility.CI_Individual or MD_Metadata.contact > CI_Responsibility.CI_Organization	M
Date(s) associated with the metadata (creation date)	MD_Metadata.dateInfo > CI_Date	M
Metadata standard name	MD_Metadata.metadataStandard > CI_Citation.title	O
Metadata standard version	MD_Metadata.metadataStandard > CI_Citation.edition	O
Service title	MD_Metadata.identificationInfo > SV_ServiceIdentification.citation > CI_Citation.title	M
Date used to identify the service	MD_Metadata.identificationInfo > SV_ServiceIdentification.citation > CI_Citation.date	M
Resource identifier	MD_Metadata.identificationInfo > SV_ServiceIdentification.citation > CI_Citation.identifier > MD_Identifier	O
Resource abstract	MD_Metadata.identificationInfo > SV_ServiceIdentification.abstract	M
Responsible party	MD_Metadata.identificationInfo > SV_ServiceIdentification.pointOfContact > CI_Responsibility	O
Spatial representation type	MD_Metadata.identificationInfo > MD_DataIdentification.spatialRepresentationType	O
Spatial resolution of the dataset	MD_Metadata.identificationInfo > MD_Identification.spatialResolution > MD_Resolution.distance or MD_Resolution.equivalentScale or MD_resolution.vertical or MD_Resolution.angularDistance or MD_Resolution.levelOfDetail	O [e]
Dataset language	MD_Metadata.identificationInfo > MD_DataIdentification.defaultLocale > PT_Locale.language	M
Dataset character set	MD_Metadata.identificationInfo > MD_DataIdentification.defaultLocale > PT_Locale.characterEncoding	C [f]
Service topic category	MD_Metadata.identificationInfo > SV_ServiceIdentification.topicCategory	M
Geographic location of the service (by four coordinates or by description)	MD_Metadata.identificationInfo > SV_ServiceIdentification.extent > EX_Extent.geographicElement > EX_GeographicExtent > EX_GeographicBoundingBox or EX_GeographicDescription	C [g, h]
Temporal extent information for the service	MD_Metadata.identificationInfo > SV_ServiceIdentification.extent > EX_Extent.temporalElement	O

Name	Path	Obligation
Vertical extent information for the dataset	MD_Metadata.identificationInfo > SV_ServiceIdentification.extent > EX_Extent.verticalElement > EX_VerticalExtent	O
Lineage	MD_Metadata.resourceLineage > LI_Lineage	O
Reference system	MD_Metadata.referenceSystemInfo > MD_ReferenceSystem.referenceSystemIdentifier > RS_Identifier	O
Distribution Format	MD_Metadata.distributionInfo > MD_Distribution > MD_Format	O
On-line link	MD_Metadata.identificationInfo > SV_ServiceIdentification.citation > CI_Citation.onlineResource > CI_OnlineResource	O
Constraints on resource access and use	MD_Metadata.identificationInfo > SV_ServiceIdentification > MD_Constraints.useLimitations and/or MD_LegalConstraints and/or MD_SecurityConstraints	O
Resource scope	MD_Metadata.metadataScope > MD_Scope.resourceScope	C [i]
Operated dataset	MD_Metadata > SV_ServiceIdentification.operatedDataset > CI_Citation	C [j]
Operates on	MD_Metadata > SV_ServiceIdentification.operatesOn > MD_Identifier	C [j]

a) the Profile imposes a mandatory obligation on the metadata element metadataIdentifier

b) language: documented if not defined by the encoding process.

c) characterEncoding: documented if UTF-8, is not used and not defined by the encoding process.

d) documented if a higher level of hierarchy level exists (for example, if the geographic ‘dataset’ is part of a ‘series’).

e) distance is preferred over equivalentScale because the scale will change when presented at different sizes on a screen. distance or equivalentScale must be documented if available.

f) characterSet: documented if UTF-8 is not used.

g) include either the geographic bounding box (extents) or the geographic description (It is recommended that geographic bounding box should be used - see Section 5.6.3).

h) if any one of west longitude, east longitude, south latitude or north latitude exists, then the remaining three must also be completed.

i) Mandatory for resources that are not datasets.

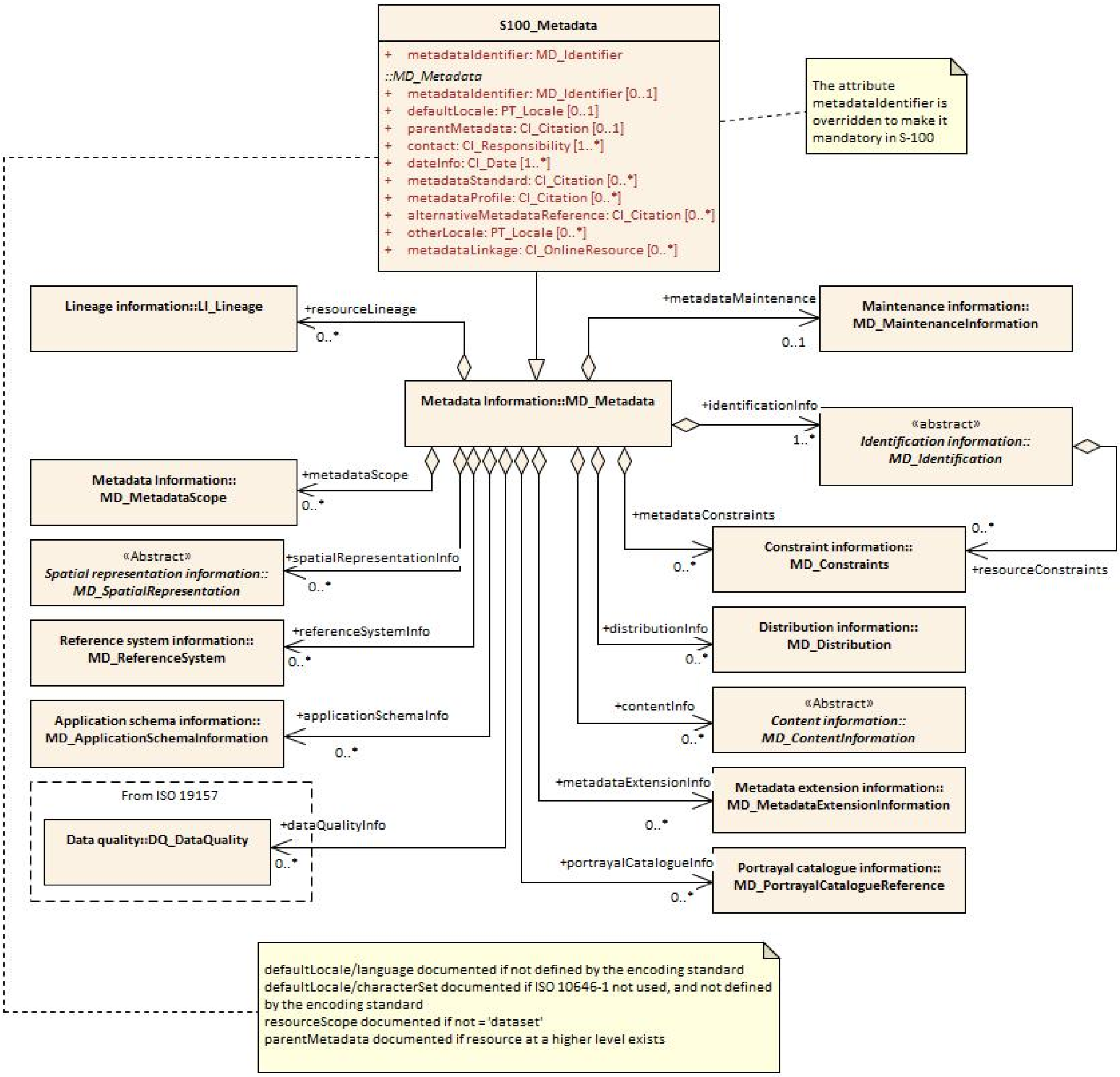
j) Reference to the resource on which the service operates. For any one resource, either ‘operated dataset’ or ‘operates on’ is used (that is, both must not be used for the same resource).

Page intentionally left blank

Appendix 4a-A
Metadata Schema Class Information
(normative)

The structure of metadata included in the S-100 Metadata Profile is defined with reference to UML diagrams that identify metadata packages and classes included in ISO 19115-1:2014 (and further modified by Amendment 1 ISO 19115-1:2018).

The new class *S100_Metadata* shows the relationship to *MD_Metadata* and its related metadata classes. For the purpose of this Profile *Metadata Schema classes* replaces the equivalent diagram Figure 4 in ISO 19115-1.



Source: Adapted from ISO 19115-1:2014

Figure 4a-A-1— Metadata Schema classes

Page intentionally left blank

Appendix 4a-B
Data Dictionary
(normative)

The data dictionary in Annex B of ISO 19115-1:2014 (and further modified by ISO 19115-1:2014/Amdt1:2018) describes the characteristics of the metadata identified in the UML package diagrams included in ISO 19115-1.

Modifications to the data dictionary, required to recognise the extension to the metadata element *metadataIdentifier* that was introduced in this Profile, are included at Table 4a-B-1. The information contained in the table replaces, or is in addition to, that provided at B.2, Annex B, ISO 19115-1:2014 and ISO 19115-1:2014/Amdt1:2018.

Table 4a-B-1 — Modifications to the data dictionary ISO 19115-1:2014

	Name / Role name	Definition	Ob	Max Occ	Data type	Comment
1	MD_Metadata	root entity which defines metadata about a resource or resources	M	1	Class	See B.2, Annex B, ISO 19115-1:2014
1.1	S100_Metadata	root entity which defines metadata about a resource or resources	M	1	Class	Specialises *MD_Metadata* class
2	metadataIdentifier	unique identifier for this metadata file	M	1	CharacterString	Free text (changed obligation from optional to mandatory)

Ob = Obligation / Condition; **Max Occ** = Maximum occurrence

Page intentionally left blank

Appendix 4a-C
Metadata Implementation
(normative)

Background

ISO 19115-1:2014 defines the content of a set of metadata elements, their definitions, data types and inherent dependencies. The logical model of the metadata specifies the content and not the form of implementation or the form of presentation. A primary goal in the management of metadata for resources is the ability to access the metadata and the related resource it describes. This requires software implementations using common encoding methods to achieve operational use of the metadata.

It is necessary to implement the Profile in order to prove compliance. ISO/TS 19115-3:2016 is an XML Schema implementation of ISO 19115-1:2014 and can be used to prove partial compliance to ISO 19115-1:2014 and the S-100 metadata profile. IHO has developed additional Schematron rules to enforce the additional restriction for the metadataIdentifier element. Proof of compliance to this profile be via validation of the XML document instances against the ISO/TS 19115-3:2016 XML Schema Definition (XSDs) and the S-100 Schematron Metadata Rules.

While the S100_Metadata class specializes the MD_Metadata class, the specialization only involves restricting metadataIdentifier from optional to mandatory. Therefore the MD_Metadata root element must be used instead of the S100_Metadata for XML instances of S-100 metadata in order to ensure interoperability with ISO standards and software tool.

Granularity of geographic data supported: The notion of cataloguing a set of related documents together in a discoverable series is common practice for map Catalogues. With digital spatial data, the definition of what constitutes a dataset is more problematic and reflects the institutional and software environments of the originating organisation. Common metadata can be derived for a series of related geographic datasets, and such metadata is generally relevant or can be inherited by each of the dataset instances. Software to support this inheritance of metadata for geographic data within a cataloguing system can simplify data entry, update and reporting.

There is a potential hierarchy of reusable metadata that can be employed in implementing a metadata collection. By creating several levels of abstraction, a linked hierarchy can assist in filtering or targeting user queries to the requested level of detail. The hierarchy should not necessarily be interpreted to require multiple copies of metadata being managed online. Conversely, the definition of general metadata can be supplemented by spatially specific metadata that, when queried, either inherits or overrides the general case.

Through the use of pointers this method can reduce the redundancy of metadata managed at a site and provide for different views of the holdings by users. These ‘pointers’ are implemented in the XSDs by XLink attributes.

Dependencies between metadata document elements and elements in other metadata documents may exist, allowing inheritance of metadata between hierarchy levels. Dependencies between metadata document elements and resources from standard registers may exist, allowing re-use of standard resources without copying the content. For either purpose the dependency may be made explicit through use of the XLink attributes which are available on most property elements in the XML representation. XLink:href is used to point to the re-used resource. XLink:arcrole is used to indicate the kind of re-use. XLink:role is used to indicate the nature of the reused resource.

Page intentionally left blank

Appendix 4a-D
Metadata Extensions
(normative)

These rules are an adaptation of the Metadata extension rules provided in Annex C of ISO 19115-1:2014. These rules are meant to be used as a common rule set for how to extend S-100 metadata, and aims to create a common process that gives predictability for implementers.

Types of extensions

The following types of extensions shall be allowed:

1) adding a new metadata package;
2) creating a new metadata codelist to replace the domain of an existing metadata element that has "free text" listed as its domain value;
3) creating new metadata codelist elements (expanding a codelist);
4) adding a new metadata element;
5) adding a new metadata class;
6) imposing a more stringent obligation on an existing metadata element; and
7) imposing a more restrictive domain on an existing metadata element.

When creating an extension

Prior to the creation of extended metadata, a careful review of the existing metadata within ISO 19115-1 must be performed to confirm that suitable metadata does not already exist. If suitable metadata exist within ISO 19115-1, then it must be used. For each extended metadata package, class, and/or element, the name, definition, obligation, condition, maximum occurrence, data type, and domain values shall be defined. Relationships shall be defined so that a structure and Schema can be determined. The relationships should be defined well enough that it is clear how extended metadata relates to the various components of S-100, including existing metadata, used to create the product where the extended metadata is used.

Rules for creating an extension

1) Extended metadata elements shall not be used to change the name, definition or data type of an existing element.
2) Extended metadata may be defined as classes and may include extended and existing metadata elements as components.
3) An extension is permitted to impose more stringent obligations on existing metadata elements than the standard requires. (Metadata elements that are optional in the standard may be mandatory in an extension.)
4) An extension is permitted to contain metadata elements with domains that are more restrictive than the standard. (Metadata elements whose domains have free text in the standard may have a closed list of appropriate values in the profile.)
5) An extension is permitted to restrict the use of domain values allowed by the standard. (If the standard contains five values in the domain of an existing metadata element, the extension may specify that its domain consists of three domain values. The extension shall require that the user select a value from the three domain values.)
6) An extension is permitted to expand the number of values in codelists or enumerated lists. Extending codelists or enumerated lists are discouraged, even in profiles. When they must be extended care should be taken to minimize the number of additional entries. Also, the extended codelist or enumerated list should be published or otherwise made available.
7) An extension shall not permit anything not allowed by S-100.
8) References to files shall be of type URI following the formatting.

Page intentionally left blank

S-100 – Part 4b

Metadata for Imagery and Gridded Data

Page intentionally left blank

Contents

Page intentionally left blank

4b-1 Scope

The general scope of parts 4a, 4b and 4c has been described at the beginning of part 4a. This part concerns itself specifically with the growing requirement to manage large volumes of imagery and gridded data which most hydrographic organizations have in addition to handling the vector data. There are many different imagery and gridded data formats and these types of datasets are often stored on distributed systems leading to problems of data discovery, management and exchange.

The production of imagery and gridded data follows the processes that usually begin with the collection of data, scanning of charts and reference documents, and other sensing methods. These types of datasets are often used for the production of paper charts, Electronic Navigational Charts (ENCs), Raster Navigational Charts, and nautical publications. Their production processes need to be documented in order to maintain quality control over the end products. Furthermore, metadata about the geometry of the measuring process and the properties of the measuring equipment needs to be retained with the raw data in order to support the production and maintenance processes.

This metadata Part of S-100 is based on ISO 19115-2:2009.

4b-2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/TS 19103, *Geographic information — Conceptual schema language*

ISO 19107:2003, *Geographic information — Spatial schema*

ISO 19115-1:2018, *Geographic information – Metadata – Part 1 – Fundamentals* (published as ISO 19115-1:2014, amended by Amendment 1, 2018)

ISO 19115-2:2009, *Geographic information - Metadata - Part 2: Extensions for imagery and gridded data*

ISO 19119:2016, *Geographic information – Services*

ISO/TS 19115-3:2016, *Geographic information - Metadata - XML schema implementation for fundamental concepts*

ISO 19157:2018, *Geographic information – Data Quality* (published as ISO 19157:2013, amended by Amendment 1, 2018)

IHO S-61 *Product Specification for Raster Navigational Charts*

4b-2.1 Informative references

The following references have been superseded by later editions or are otherwise useful though not normative:

ISO 19115:2005, *Geographic information — Metadata*

ISO/TS 19139, *Geographic information — Metadata — XML schema implementation*

ISO 19119:2005, *Geographic information – Services*

4b-3 Imagery and gridded data metadata

ISO 19115-1 identifies the metadata required to describe digital geographic data, and the extensions described in this section identify the metadata required to describe digital geospatial imagery and gridded data. Digital geospatial imagery and gridded metadata may also be provided for aggregations of datasets.

4b-3.1 Associated ISO standards

ISO 19115-1 is designed to be the general metadata standard applicable to all geographic data sets. It identifies a set of core metadata derived from the many metadata elements and also specifies the conditions under which they should be used (that is mandatory, conditional, or optional). Although there is some service metadata in ISO 19115-1, (particularly in the area of identification), much of the service metadata is defined in ISO 19119 (Services). ISO 19115 makes provision for limited metadata describing spatial and temporal Schemas.

ISO 19115-2 extends the metadata defined in ISO 19115 and identifies additional metadata (such as data quality, spatial representation, content, and acquisition information), required to describe imagery and gridded data. It provides information about the properties of the measuring equipment used to acquire data, geometry of the measuring processes employed by the equipment, and production processes used to digitize the raw data.

Geolocation information is a very important metadata component required for imagery. ISO 19115-1 and 19115-2 may not include sufficient geolocation metadata for imagery and gridded data. It may therefore be necessary to reference ISO 19130. This standard specifies additional information required to support geolocation and also defines how sensor measurements and geolocation information are logically associated. The georeferencing information in ISO 19130 is a subset of that described in ISO 19115-2. In order to develop a full set of imagery metadata, it may be necessary to combine the relevant sections from ISO 19115-1 and 19115-2, with the geolocation information or sensor properties from ISO 19130.

ISO 19115-3 - XML Schema implementation for fundamental concepts, expands ISO 19115-1 and 19115-2 by defining new constraint types that further refine the metadata elements for implementation. It also defines the rules used for deriving an XML Schema from the ISO abstract UML models.

4b-3.2 Metadata packages

The relationships between the packages contained in ISO 19115-1 and the extensions for geospatial imagery and gridded data are illustrated in Figure 4b-1 below. Dependencies on other packages are also shown in the figure. ISO 19115-2 packages are shown with no fill, ISO 19115-1 packages with grey fill, and the others (ISO 19107 (Geometry), ISO 19157 (Data quality) and ISO 19103 (Conceptual Schema language)) in other colours. These metadata extensions have been fully documented using both UML models and a data dictionary, in ISO/TC211 19115–2 - Annex A and Annex B respectively.

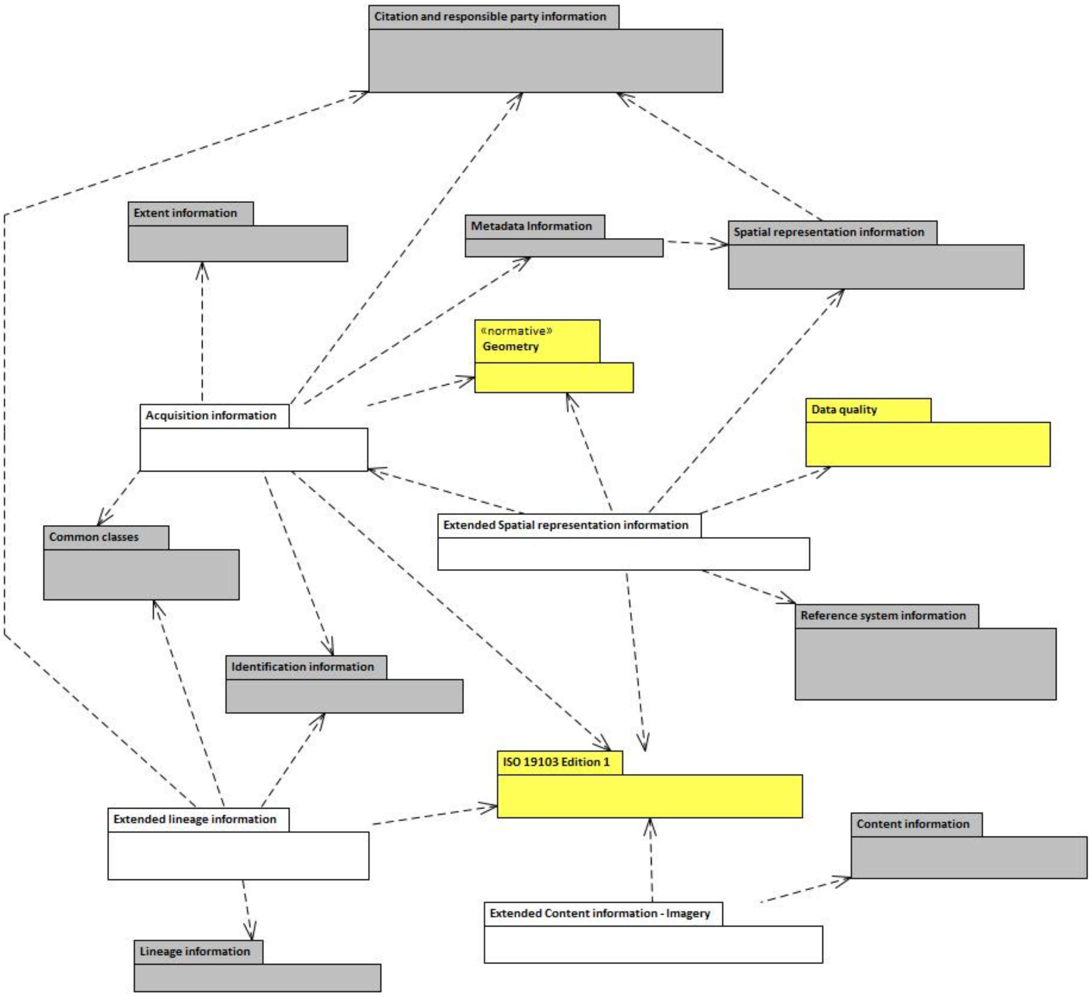


Figure 4b-1 — Metadata packages (from ISO 19115-2:2018)

It should also be noted that, to ensure global uniqueness, ISO/TS 19103 requires that all class names must be defined by a bi-alpha prefix that identifies the package to which a class belongs. The ISO 19115 series uses the prefixes MD (Metadata), CI (Citation), DQ (Data quality), EX (Extent), and LI (Lineage). To differentiate between classes used in ISO 19115-1 and those used in ISO 19115-2 (Extensions for imagery and gridded data), the MI prefix is used for imagery and gridded metadata, and LE and QE are used for extended Lineage and Data quality classes respectively. (Data quality classes are now defined in ISO 19157.) Table 4b-1 contains the list of package identifiers for the classes used for metadata.

Table 4b-1 —UML Package Identifiers

Identifier	Information Type	Standard
MD	Metadata	ISO 19115-1
MI	Metadata for Imagery	ISO 19115-2
DQ	Data Quality	ISO 19157
QE	Data quality Extended	ISO 19115-2
CI	Citation	ISO 19115-1
LI	Lineage	ISO 19115-1
LE	Lineage Extended for Imagery	ISO 19115-2
EX	Extent	ISO 19115-1
GM	Geometry	ISO 19107
MX	Metadata – XML Schema	ISO/TS 19139

PT	Polylinguistic Text	ISO/TS 19103
RS	Reference System	ISO 19115-1
SC	Spatial Coordinates	ISO 19111
SV	Metadata for services	ISO 19115-1

4b-3.2.1 Metadata Entity Set for Imagery

MI_Metadata is a subclass of MD_Metadata which aggregates the optional entity MI_AcquisitionInformation. See sections A.2.1 and B.2.1 of ISO 19115-2 for additional descriptive information and the data dictionary respectively.

4b-3.2.2 Data quality information for Imagery

Information about the sources and production processes used in producing an imagery or gridded dataset has been included in an additional Data Quality for Imagery package, as ISO 19115-1 only makes provision for a general assessment of the quality. The following additional classes are listed below. A more detailed description of the classes and associated data dictionary are provided in ISO 19115-2, sections A.2.2 and B.2.2 respectively, and ISO 19157.

1) QE_CoverageResult is a specified subclass of DQ_Result and aggregates information required to report data quality for a coverage. It is based on concepts from ISO 19115 and ISO 19139.
2) QE_Usability is a specified subclass of DQ_Element. It is intended to provide user specific.
3) Quality information about a dataset’s suitability for a particular application.
4) LE_ProcessStep is a specified subclass of LI_ProcessStep and contains additional information on the history of the algorithms used and processing performed to produce the data. LE_ProcessStep aggregates the following entities:
   a) LE_Processing, describes the procedure (such as software used, parameters, and processing documentation) by which the algorithm was applied to generate the data from the source data. LE_Processing aggregates LE_Algorithm, which describes the methodology used to derive the data from the source data;
   b) LE_ProcessStepReport identifies external information describing the processing of the data;
   c) LE_Source is a specified subclass of LI_Source and describes the output of a process step.

4b-3.2.3 Spatial representation information for Imagery

This package contains information concerning the mechanisms used to represent spatial information. This package defines the following entities:

1) MI_Georectified contains check point information to further specify georectification details of the imagery or gridded data. It aggregates MI_GCP;
2) MI_Georeferenceable makes provision for the inclusion of additional information that can be used to geolocate the data. It aggregates MI_GeolocationInformation.

4b-3.2.4 Content information for Imagery

Although this package is part of ISO 19115-1, the following entities have been included to better cater for imagery and gridded data:

1) MI_Band (subclass of MD_Band) - defines additional attributes for specifying properties of individual wavelength bands in an imagery and gridded dataset;
2) MI_ImageDescription (subclass of MD_ImageDescription) - used to aggregate MI_RangeElementDescription;
3) MI_CoverageDescription (subclass of MD_CoverageDescription) - used to aggregate MI_RangeElementDescription;
4) MI_RangeElementDescription - used to provide range elements used in a coverage dataset.

4b-3.2.5 Acquisition Information for Imagery

MI_AcquisitionInformation is an aggregate of the following entities:

1) MI_Instrument (the measuring instruments used to acquire the data);
2) MI_Operation,(the overall data gathering program to which the data contribute);
3) MI_Platform (the platform from which the data were taken);
4) MI_Objective (the characteristics and geometry of the intended object to be observed);
5) MI_Requirement (the user requirements used to derive the acquisition plan);
6) MI_Plan (the acquisition plan that was implemented to acquire the data).

Two additional classes are required to provide information on the acquisition of the data. These are:

1) MI_Event describes a significant event that occurred during data acquisition. An event can be associated with an operation, objective, or platform pass; and
2) MI_PlatformPass identifies a particular pass made by the platform during data acquisition. A platform pass is used to provide supporting identifying information for an event and for data acquisition of a particular objective.

4b-4 UML diagrams and data dictionary

The metadata Schemas for the imagery and gridded data are included in ISO 19115–2 (Annex A) in the form of UML class diagrams. These diagrams augment the UML diagrams shown in ISO 19115-1.

ISO 19115–2, Annex B contains the element and entity definitions for the metadata Schemas defined in Annex A. The dictionary, in conjunction with the diagrams presented in Annex A and in combination with the UML diagrams and data dictionary presented in ISO 19115-1, serves to fully define the total abstract model for metadata.

Enumerations and their values provided in ISO 19115-1 are normative. User extensions to enumerations shall follow the rules as described in ISO 19115-1 and Annex 4a-E of S-100.

Page intentionally left blank

S-100 – Part 4c

Metadata - Data Quality

Page intentionally left blank

Contents

Page intentionally left blank

4c-1 Scope

The general scope of Parts 4a, b and c has been described at the beginning of Part 4a. This Part is a metadata quality guidance and incorporates quality measures as described in ISO 19113, 19114 and 19138 and complies with ISO 19106 *Geographical Information – Profiles* which describes the rules for developing profiles of the 19100 series standards. This guidance is applicable to IHO hydrographic data sets, data set series, and individual features and feature properties. It is intended for hydrographic requirements and describes how to document information about the quality of digital geographic data.

The purpose of this Part is to:

1) Provide data producers with appropriate information to characterize their geographic data properly;
2) Enable users to determine whether geographic data in a holding will be of use to them.

It defines:

1) Mandatory and conditional metadata sections, metadata entities and metadata elements;
2) Optional metadata elements to allow for more detailed description of geographic data.

Although this document is largely based on the standards mentioned above, additional standards are referenced where relevant. (See section 4c-2 References).

4c-2 References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this metadata guidance.

ISO 19104, *Geographic information – Terminology*

ISO 19106, *Geographic information — Profiles*

ISO 19107, *Geographic information — Spatial schema*

ISO 19108, *Geographic information — Temporal schema*

ISO 19115:2003, *Geographic information — Metadata*

ISO 19113, *Geographic information — Quality principles*

ISO 19114, *Geographic information — Quality evaluation procedures*

ISO 19138, *Geographic information – Quality measures*

ISO 19139 *Geographic information – Metadata – XML schema implementation* (preliminary Draft Technical Specification).

ISO 639, *Code for the representation of names of languages*

ISO 3166-1, *Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes*

ISO 8601:2000, *Data elements and interchange formats -- Information interchange -- Representation of dates and times*

ISO 639-1:2002, *Codes for the representation of names of languages - Part 1: Alpha-2 code*

ISO 639-2:1998, *Codes for the representation of names of languages -- Part 2: Alpha-3 code*

4c-3 Content

ISO 19115 defines almost 300 metadata elements, which include a group of core metadata elements. S-100 Part 4c (Metadata) describes in general how these are used within S-100. However, to fully describe hydrographic data additional elements are needed. This document describes elements for quality measures as defined and described in ISO 19138.

4c-3.1 ISO 19138 Quality Measures and UML Classes

The IHO Quality Metadata Guidance contains optional quality metadata elements for hydrographic requirements. Additional 19115 elements are available for use; however they may not be recognised by systems not conforming to this profile. The metadata packages used in this profile are shown in Unified Modeling Language (UML) class diagrams at Appendix 4c-A.

S-100 Quality Measure class structure is derived from ISO 19115 Geographic Information Metadata. The attributes described in the S-100 Quality classes each correspond to independent quality measures. Full descriptions of these measures are contained in ISO 19138 Geographic Information Data Quality Measures.

All of the S-100 Quality measures are intentionally optional so that different measures may be used for different types of data. Where multiple attributes describe the same measure in different ways, either only one measure should be used or the measures must be described in a consistent manner.

Additional quality measures may be described in a register of quality measures as described in ISO 19138 Annex B.

4c-3.2 Core Metadata

Core metadata elements are described in S-100 Part 4a. Dataset and feature quality metadata can be linked to a higher hierarchy level field, and all these levels may be supplied in one file or separate metadata files.

Appendix 4c-A
Hydrographic Quality Metadata profile, UML Diagrams
(informative)

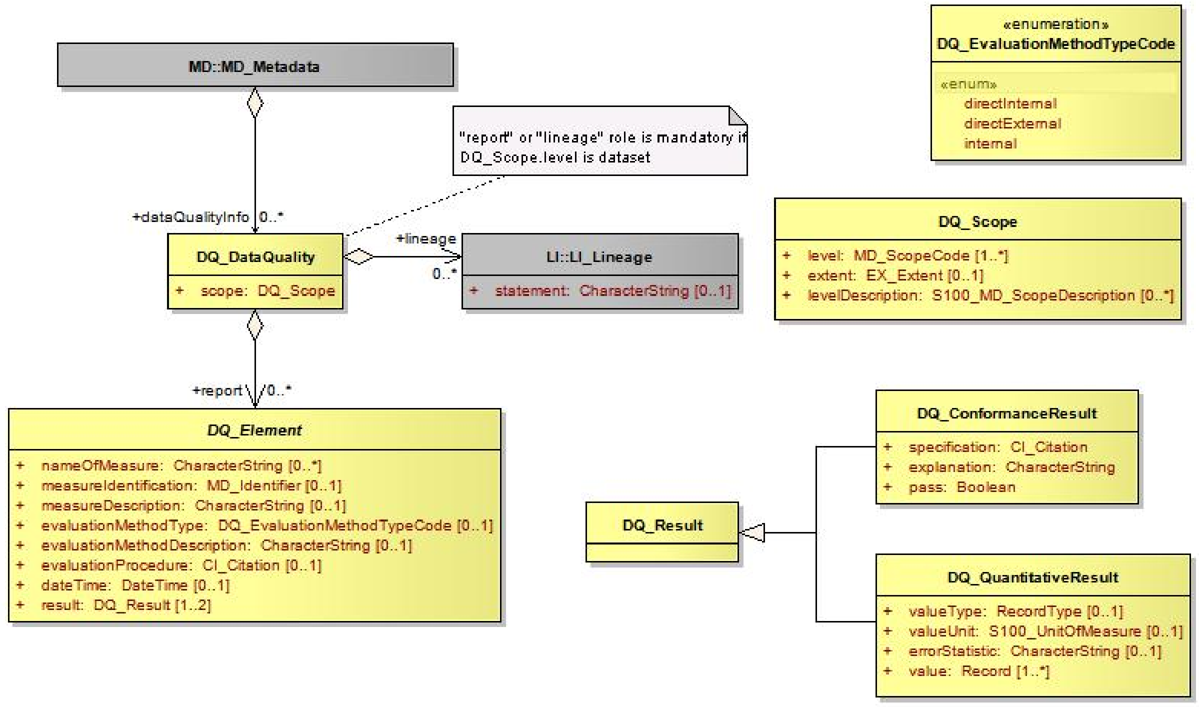


Figure 4c-A-1— Data Quality UML (from ISO 19115)

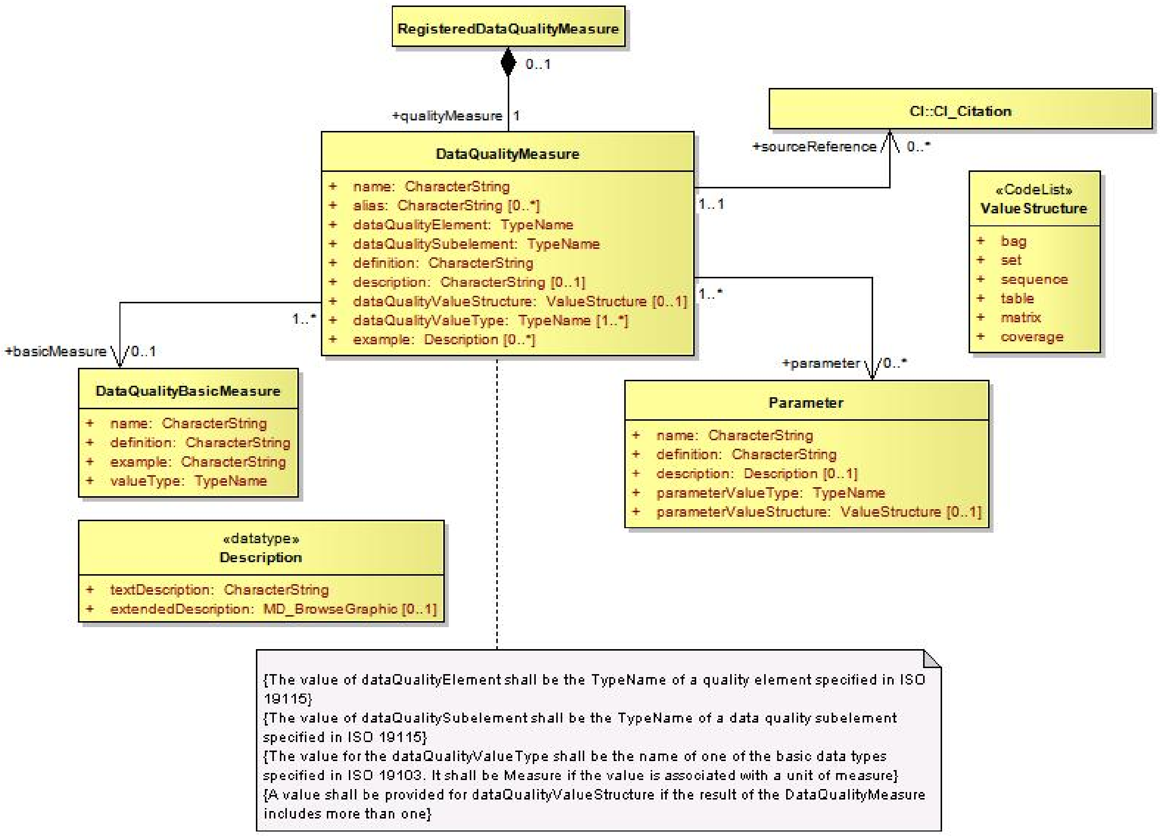


Figure 4c-A-2 — Data Quality Measure Registry UML (from ISO 19138)

Page intentionally left blank

Appendix 4c-B
Hydrographic Quality Metadata profile Data Dictionary
(normative)

The hydrographic metadata Catalogue table below has been derived from the ISO 19115 Standard.

The table contains the following information:

1) The first column "*ISO LineNo*." refers to the line numbers in the ISO 19115 Standard, however as this profile does not use all the 19115 elements, line numbers may not always be contiguous.
2) Name/role name is a label assigned to a metadata entity or to a metadata element. Further columns could give the name or meaning in other languages.
3) Definition column provides a description of the metadata entity/element.
4) The obligation descriptor provides an indication of whether a metadata entity or metadata element shall always be documented or will only sometimes be documented. This descriptor may have the following values: M (mandatory), C (conditional), or O (optional).
5) The Occurrence column specifies the maximum number of instances the metadata entity or the metadata element may have. Single occurrences are shown by "1"; repeating occurrences are represented by "N". Fixed number occurrences other than one are allowed, and will be represented by the corresponding number (that is "2", "3"…etc.).
6) Data type specifies a set of distinct values for representing the metadata elements; for example, integer, real, string, DateTime, and Boolean. The data type attribute is also used to define metadata entities, stereotypes, and metadata associations.
7) Domain - for an entity, the domain indicates the line numbers covered by that entity.

Page intentionally left blank

ISO Line No.	Name / role name	Definition	Obligation	Maximum Occurrence	Data Type	Domain
	B.2.4 Data quality information					
	B.2.4.1 General					
78	DQ_DataQuality	Quality information for the data specified by a data quality scope	Use obligation from referencing object	Use maximum occurrence from referencing object	Aggregated Class (MD_Metadata)	Lines 79-81
79	scope	The specific data to which the data quality information applies	M	1	Class	DQ_Scope <<DataType>> (B.2.4.4)
80	Role name: report	Quantitative quality information for the data specified by the scope	C / lineage not provided?	N	Association	DQ_Element <<Abstract>> (B 2.4.2)
81	Role name: lineage	Non-quantitative quality information about the lineage of the data specified by the scope	C / report not provided?	1	Association	LI_Lineage (B 2.4.1)
	B.2.4.2 Lineage information					
	B.2.4.2.1 General					
82	LI_Lineage	Information about the events or source data used in constructing the data specified by the scope or lack of knowledge about lineage	Use obligation from referencing object	Use maximum occurrence from referencing object	Aggregated Class (DQ_DataQuality)	Lines 83-85
83	statement	General explanation of the data producer's knowledge about the lineage of a dataset	C / (DQ_DataQuality.scope.DQ_Scope.level = "dataset" or "series")?	1	CharacterString	Free text
84	Role name: processStep	Information about an event in the creation process for the data specified by the scope	C / mandatory if statement and source not provided?	N	Association	LI_ProcessStep (B.2.4.1.1)
85	Role name: source	Information about the source data used in creating the data specified by the scope	C / mandatory if statement and processStep not provided?	N	Association	LI_Source (B.2.4.1.2)

ISO Line No.	Name / role name	Definition	Obligation	Maximum Occurrence	Data Type	Domain
	B.2.4.2.2 Process step information					
86	LI_ProcessStep	Information about an event in the creation process for the data specified by the scope	Use obligation from referencing object	Use maximum occurrence from referencing object	Aggregated Class (LI_Lineage)	Lines 86-91
87	description	Description of the event, including related parameters or tolerances	M	1	CharacterString	Free Text
88	rationale	Requirement or purpose for the process step	O	1	CharacterString	Free Text
89	dateTime	Date and time or range of date and time on or over which the process step occurred	O	1	Class	DateTime (B.4.2)
90	processor	Identification of, and means of communication with, person(s) and organisation(s) associated with the process step	O	N	Class	CI_ResponsibleParty <<DataType>> (B.3.2)
91	Role name: source	Information about the source data used in creating the data specified by the scope	O	N	Association	LI_Source (B.2.4.1.2)
	B.2.4.2.3 Source information					
92	LI_Source	Information about the source data used in creating the data specified by the scope	Use obligation from referencing object	Use maximum occurrence from referencing object	Aggregated Class (LI_Lineage)	Lines 93-98
93	description	Detailed description of the level of the source data	C/ sourceExtent not provided?	1	CharacterString	Free Text
94	scaleDenominator	Denominator of the representative fraction on a source map	O	1	Class	MD_RepresentativeFraction <<DataType>> (B.2.2.3)
95	sourceReferenceSystem	Spatial reference system used by the source data	O	1	Class	MD_ReferenceSystem (B.2.7)
96	sourceCitation	Recommended reference to be used for the source data	O	1	Class	CI_Citation <<DataType>> (B.3.2)
97	sourceExtent	Information about the spatial, vertical and temporal extent of the source data	C/ description not	N	Class	EX_Extent <<DataType>> (B.3.1)

ISO Line No.	Name / role name	Definition	Obligation	Maximum Occurrence	Data Type	Domain
			provided?			
98	Role name: sourceStep	Information about an event in the creation process for the source data	O	N	Association	LI_ProcessStep (B.2.4.1.1)
	B.2.4.2 Data quality element information					
99	DQ_Element	Type of test applied to the data specified by a data quality scope	Use obligation from referencing object	Use maximum occurrence from referencing object	Aggregated Class	Lines 100-107
100	nameOfMeasure	Name of the test applied to the data	O	N	CharacterString	Free text
101	measureIdentification	Code identifying a registered standard procedure	O	1	Class (19138 List)	MD_Identifier.IHO_DqMeasure <<DataType>> (B.2.7.2)
102	measureDescription	Description of the measure being determined	O	1	CharacterString	Free text
103	evaluationMethodType	Type of method used to evaluate quality of the dataset	O	1	Class	DQ_EvaluationMethodTypeCode << Enumeration >> (B.5.6)
104	evaluationMethodDescription	Description of the evaluation method	O	1	CharacterString	Free text
105	evaluationProcedure	Reference to the procedure information	O	1	Class	CI_Citation <<DataType>> (B 3.2)
106	dateTime	Date or range of dates on which a data quality measure was applied	O	1	Class	DateTime (B.4.2)
107	result	Value (or set of values) obtained from applying a data quality measure or the out come of evaluating the obtained value (or set of values) against a specified acceptable conformance quality level	M	2	Class	DQ_Result <<DataType>> (B.2.4.3)

Page intentionally left blank

Appendix 4c-C
Hydrographic Quality Metadata Attribute Definitions

DQ_AbsoluteExternalPositionalAccuracy

Closeness of reported coordinative values to values accepted as or being true. [Per ISO 19115]

Public Attributes:

meanValuePositionalUncertainties[0..1] : Real

Mean value of the positional uncertainties for a set of positions where the positional uncertainties are defined as the distance between a measured position and what is considered as the corresponding true position. [Adapted from ISO 19138]

meanExcludingOutliers[0..1] : Real

Mean value of the positional uncertainties, excluding outliers. For a set of points where the distance does not exceed a defined threshold, the arithmetical average of distances between their measured positions and what is considered as the corresponding true positions. [Adapted from ISO 19138]

numberOfPositionalUncertaintiesAboveThreshold[0..1] : Integer

Number of positional uncertainties above a given threshold for a set of positions. The errors are defined as the distance between a measured position and what is considered as the corresponding true position. [Adapted from ISO 19138]

rateOfPositionalErrorsAboveThreshold[0..1] : Real

Number of positional uncertainties above a given threshold for a set of positions in relation to the total number of measured positions. The errors are defined as the distance between the measured position and what is considered as the corresponding true position. [Adapted from ISO 19138]

covarianceMatrix[0..1] : Real Matrix

Symmetrical square matrix with variances of point coordinates on the main diagonal and covariances between these coordinates as off diagonal elements. [Adapted from ISO 19138]

linearErrorProbable[0..1] : Real

Half length of the interval defined by an upper and lower limit in which the true value lies with probability 50%. [Adapted from ISO 19138]

standardLinearError[0..1] : Real

Half length of the interval defined by an upper and lower limit in which the true value lies with probability 68.3%. [Adapted from ISO 19138].

linearMapAccuracy2Sigma[0..1] : Real

Half length of the interval defined by an upper and lower limit in which the true value lies with probability 90%. [Adapted from ISO 19138].

linearMapAccuracy3Sigma[0..1] : Real

Half length of the interval defined by an upper and lower limit in which the true value lies with probability 95%. [Adapted from ISO 19138].

linearMapAccuracy4Sigma[0..1] : Real

Half length of the interval defined by an upper and lower limit in which the true value lies with probability 99%. [Adapted from ISO 19138].

nearCertainityLinearError[0..1] : Real

Half length of the interval defined by an upper and lower limit in which the true value lies with probability 99.8%. [Adapted from ISO 19138].

RMSError[0..1] : Real

Standard deviation where the true value is not estimated from the observations but known apriori. [Adapted from ISO 19138].

circularStandardDeviation[0..1] : Real

Radius describing a circle in which the true point location lies with the probability of 39.4%. [Adapted from ISO 19138].

circularErrorProbable[0..1] : Real

Radius describing a circle in which the true point location lies with the probability of 50%. [Adapted from ISO 19138].

circularMapAccuracyStandard[0..1] : Real

Radius describing a circle in which the true point location lies with the probability of 90%. [Adapted from ISO 19138].

circularError95[0..1] : Real

Radius describing a circle in which the true point location lies with the probability of 95%. [Adapted from ISO 19138].

circularNearCertaintyError[0..1] : Real

Radius describing a circle in which the true point location lies with the probability of 99.8%. [Adapted from ISO 19138].

RMSErrorPlanimetry[0..1] : Real

Radius of a circle around a given point in which the true value lies with true value P. [Adapted from ISO 19138].

CMASError[0..1] : Real

The absolute horizontal accuracy of the data's coordinates expressed in terms of circular error at 90% probability given that a bias is present, per the equation in table D.48 in ISO 19138. [Adapted from ISO 19138].

ACE_CE90[0..1] : Real

The absolute horizontal accuracy of the data's coordinates expressed in terms of circular error at 90% probability given that a bias is present, per the equation in table D.49 in ISO 19138. [Adapted from ISO 19138].

uncertaintyEllipse[0..1] : Record

A 2D ellipse with the two main axes indicating the direction and magnitude of the highest and lowest uncertainty of a 2D point. The data values are a record of real numbers corresponding to "phi" the bearing of the major semi-axis, and "a" and "b" the length of the two axes, per the equations in Table D.50 of ISO 19138. [Adapted from ISO 19138].

confidenceEllipse[0..1] : Record

A 2D ellipse with the two main axes indicating the direction and magnitude of the highest and lowest uncertainty of a 2D point. The data values are a record of real numbers corresponding to "phi" the bearing of the major semi-axis, and "a" and "b" the length of the two axes, per the equations in Table D.51 of ISO 19138 and a significance level parameter. [Adapted from ISO 19138].

DQ_AccuracyOfATimeMeasurement

Correctness of the temporal references of an item (reporting of error in time measurement). [Per ISO 19115]

Public Attributes:

attributeValueUncertaintyMean[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 50%. [Adapted from ISO 19138]

attributeValueUncertainty1Sigma[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 68.3%. [Adapted from ISO 19138]

attributeValueUncertainty2Sigma[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 90%. [Adapted from ISO 19138]

attributeValueUncertainty3Sigma[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 95%. [Adapted from ISO 19138]

attributeValueUncertainty4Sigma[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 99%. [Adapted from ISO 19138]

attributeValueUncertainty5Sigma[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 99.8%. [Adapted from ISO 19138]

DQ_CompletenessCommission

Excess data present in a data set. [Per ISO 19115]

Public Attributes:

excessItem[0..1] : Boolean

This data quality measure indicates that an item is incorrectly present in the data. [Adapted from ISO 19138]

This is a Boolean where TRUE indicates that the item is in excess.

numberOfExcessItems[0..1] : Integer

This data quality measure indicates the number of items in the dataset, that should not have been in the dataset. [Adapted from ISO 19138]

This is an INTEGER count of the number of excess items.

rateOfExcessItems[0..1] : Real

This data quality measure indicates the number of excess items in the dataset in relation to the number of items that should have been present. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 5 measured values and 4 valid values then the ratio is 5/4 and the reported rate = 1.25.

numberOfDuplicateFeatureInstances[0..1] : Integer

This data quality measure indicates the total number of exact duplications of feature instances within the data. This is a count of all items in the data that are incorrectly extracted with duplicate geometries. [Adapted from ISO 19138]

This is an integer representing the error count.

DQ_CompletenessOmission

This data absent from a data set. [Per ISO 19115]

Public Attributes:

missingItem[0..1] : Boolean

This data quality measure is an indicator that shows that a specific item is missing in the data. [Adapted from ISO 19138]

This is a Boolean where TRUE indicates that an item is missing.

numberOfMissingItems[0..1] : Integer

This data quality measure indicates the count of all items that should have been in the dataset and are missing. [Adapted from ISO 19138]

This is an INTEGER count of the number of missing items.

rateOfMissingItems[0..1] : Real

This data quality measure indicates the number of missing items in the dataset in relation to the number of items that should have been present. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 3 measured values and 5 values are required the ratio is 3/.5 and the reported rate = 0.6.

DQ_ConceptualConsistancy

Adherence to the rules of a Conceptual Schema. [Per ISO 19115]

Public Attributes:

conceptualSchemaNonCompliance[0..1] : Boolean

This data quality measure is an indication that an item is not compliant to the rules of the relevant Conceptual Schema. [Adapted from ISO 19138]

This is a Boolean where TRUE indicates that an item is not compliant with the rules of the Conceptual Schema.

conceptualSchemaCompliance[0..1] : Boolean

This data quality measure is an indication that an item complies with the rules of the relevant Conceptual Schema. [Adapted from ISO 19138]

This is a Boolean where TRUE indicates that an item is in compliance with the rules of the Conceptual Schema.

numberOfNonCompliantItems[0..1] : Integer

This data quality measure is a count of all items in the dataset that are noncompliant to the rules of the Conceptual Schema. If the Conceptual Schema explicitly or implicitly describes rules, these rules have to be followed. Violations against such rules, for example; can be invalid placement of features within a defined tolerance, duplication of features and invalid overlap of features. [Adapted from ISO 19138]

This is an integer count.

numberOfInvalidSurfaceOverlaps[0..1] : Integer

This data quality measure is a count of the total number of erroneous overlaps within the data. Which surfaces may overlap and which must not is application dependent. Not all overlapping surfaces are necessarily erroneous. When reporting this data quality measure the types of feature classes corresponding to the illegal overlapping surfaces have to be reported as well. [Adapted from ISO 19138]

The allowable topological levels are described in the IHO/DGIWG joint profile of ISO 19107 Geographic Information Spatial Schema. Which particular topological structure may be used with a specific dataset is defined in the Product Specification for that type of data product, for example "Chain Node Topology" for IHO S-101.

This is an error count.

nonComplianceRate[0..1] : Real

This data quality measure indicates the number of items in the dataset that are noncompliant to the rules of the Conceptual Schema in relation to the total number of these items that are expected to be in the dataset. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 5 items that are non compliant and there are 100 of the items in the dataset then the ratio is 5/100 and the reported rate = 0.05.

complianceRate[0..1] : Real

This data quality measure indicates the number of items in the dataset that are in compliance with the rules of the Conceptual Schema in relation to the total number of these items that are expected to be in the dataset. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 95 items that are compliant and there are 100 of the items in the dataset then the ratio is 95/100 and the reported rate = 0.95.

DQ_DomainConsistancy

Adherence of the values to the value domains. [Per ISO 19115]

Public Attributes:

valueDomainNonConformance[0..1] : Boolean

This data quality measure is an indication that an item is not in conformance with its value domain. [Adapted from ISO 19138]

This is a Boolean where TRUE indicates that an item is not in conformance with its value domain.

valueDomainConformance [0..1] : Boolean

This data quality measure is an indication that an item is conforming to its value domain. [Adapted from ISO 19138]

This is a Boolean where TRUE indicates that an item conforming to its value domain.

numberOfNonconformantItems[0..1] : Integer

This data quality measure is a count of all items in the dataset that are not in conformance with their value domain. [Adapted from ISO 19138]

This is an integer count.

valueDomainConformanceRate[0..1] : Real

This data quality measure indicates the number of items in the dataset that are in conformance with their value domain in relation to the total number of items in the dataset. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 95 items that are in conformance and there are 100 of the items in the dataset then the ratio is 95/100 and the reported rate = 0.95.

valueDomainNonConformanceRate[0..1] : Real

This data quality measure indicates the number of items in the dataset that are not in conformance with their value domain in relation to the total number of items in the dataset. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 5 items that are in conformance and there are 100 of the items in the dataset then the ratio is 5/100 and the reported rate = 0.05.

DQ_FormatConsistancy

Degree to which data is stored in accordance with the physical structure of the data set. [Per ISO 19115]

Public Attributes:

physicalStructureConflicts[0..1] : Integer

This data quality measure is a count of all items in the dataset that are stored in conflict with the physical structure of the dataset. [Adapted from ISO 19138]

This is an integer count.

physicalStructureConflictRate[0..1] : Real

This data quality measure indicates the number of items in the dataset that are stored in conflict with the physical structure of the dataset divided by the total number of items. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 3 items that are in conflict and there are 100 of the items in the dataset then the ratio is 3/100 and the reported rate = 0.03.

DQ_GriddedDataPositionalAccuracy

Closeness of gridded data position values to values to values accepted as or being true. [Per ISO 19113]

Public Attributes:

circularStandardDeviation[0..1] : Real

Radius describing a circle in which the true point location lies with the probability of 39.4%. [Adapted from ISO 19138]

circularErrorProbable[0..1] : Real

Radius describing a circle in which the true point location lies with the probability of 50%. [Adapted from ISO 19138]

circularMapAccuracyStandard[0..1] : Real

Radius describing a circle in which the true point location lies with the probability of 90%. [Adapted from ISO 19138]

circularError95[0..1] : Real

Radius describing a circle in which the true point location lies with the probability of 95%. [Adapted from ISO 19138]

circularNearCertaintyError[0..1] : Real

Radius describing a circle in which the true point location lies with the probability of 99.8%. [Adapted from ISO 19138]

RMSErrorPlanimetry[0..1] : Real

Radius of a circle around a given point in which the true value lies with true value P. [Adapted from ISO 19138]

CMASError[0..1] : Real

The absolute horizontal accuracy of the data's coordinates expressed in terms of circular error at 90% probability given that a bias is present, per the equation in table D.48 in ISO 19138. [Adapted from ISO 19138]

ACE_CE90[0..1] : Real

The absolute horizontal accuracy of the data's coordinates expressed in terms of circular error at 90% probability given that a bias is present, per the equation in table D.49 in ISO 19138. [Adapted from ISO 19138]

uncertaintyEllipse[0..1] : Record

A 2D ellipse with the two main axes indicating the direction and magnitude of the highest and lowest uncertainty of a 2D point. The data values are a record of real numbers corresponding to "phi" the bearing of the major semi-axis, and "a" and "b" the length of the two axes, per the equations in Table D.50 of ISO 19138. [Adapted from ISO 19138]

confidenceEllipse[0..1] : Record

A 2D ellipse with the two main axes indicating the direction and magnitude of the highest and lowest uncertainty of a 2D point. The data values are a record of real numbers corresponding to "phi" the bearing of the major semi-axis, and "a" and "b" the length of the two axes, per the equations in Table D.51 of ISO 19138 and a significance level parameter. [Adapted from ISO 19138]

DQ_NonQuantitativeAttributeAccuracy

Correctness of non-quantitative attribute. [Per ISO 19115]

Public Attributes:

numberOfIncorrectAttributeValues[0..1] : Integer

This data quality measure is count of the total number of erroneous attribute values within the relevant part of the dataset. It is a count of all attribute values where the value is incorrect. [Adapted from ISO 19138]

rateOfCorrectAttributeValues[0..1] : Real

This data quality measure indicates the number of correct attribute values in relation to the total number of attribute values. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 97 correct attribute values and there are 100 attribute values in total in the dataset then the ratio is 97/100 and the reported rate = 0.97.

rateOfIncorrectAttributeValues[0..1] : Real

This data quality measure indicates the number of attribute values where incorrect values are assigned in relation to the total number of attribute values. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 3 incorrect attribute values and there are 100 attribute values in total in the dataset then the ratio is 3/100 and the reported rate = 0.03

S100_QualityMetadata

DQ_QuantitativeAttributeAccuracy

Accuracy of a quantitative attribute. [Per ISO 19115]

Public Attributes:

attributeValueUncertaintyMean[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 50%. [Adapted from ISO 19138]

attributeValueUncertainty1Sigma[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 68.3%. [Adapted from ISO 19138]

attributeValueUncertainty2Sigma[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 90%. [Adapted from ISO 19138]

attributeValueUncertainty3Sigma[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 95%. [Adapted from ISO 19138]

attributeValueUncertainty4Sigma[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 99%. [Adapted from ISO 19138]

attributeValueUncertainty5Sigma[0..1] : Real

This data quality measure indicates the attribute value of uncertainty where half the length of the interval defined by an upper and lower limit in which the true value for the quantitative attribute lies with a probability of 99.8%. [Adapted from ISO 19138]

DQ_RelativeInternalPositionalAccuracy

Closeness of the relative positions of features in a dataset to their respective relative positions accepted as or being true. [Per ISO 19115]

Public Attributes:

relativeVerticalError[0..1] : Real

An evaluation of the random errors of one relief feature to another in the same data set or on the same map/chart. It is a function of the random errors in the two elevations with respect to a common vertical datum. [Adapted from ISO 19138]

relativeHorizontalError[0..1] : Real

An evaluation of the random errors in the horizontal position of one feature to another in the same data set or on the same map/chart. [Adapted from ISO 19138]

DQ_TemporalConsistancy

Correctness of ordered events or sequences, if reported. [Per ISO 19115]

Public Attributes:

temporalConsistencyStatement[0..1] : CharacterString

This is a qualitative statement of the consistency of the time measurement. There is no qualitative measure provided for this data quality sub-element. [Adapted from ISO 19138]

DQ_TemporalValidity

Validity of data with respect to time. [Per ISO 19115]

Public Attributes:

valueDomainNonConformance[0..1] : Boolean

This data quality measure is an indication that an item is not in conformance with its value domain. [Adapted from ISO 19138]

This is a Boolean where TRUE indicates that an item is not in conformance with its value domain.

valueDomainConformance[0..1] : Boolean

This data quality measure is an indication that an item is conforming to its value domain. [Adapted from ISO 19138]

This is a Boolean where TRUE indicates that an item is conforming to its value domain.

numberOfNonConformantItems[0..1] : Integer

This data quality measure is a count of all items in the dataset that are not in conformance with their value domain. [Adapted from ISO 19138]

This is an integer count.

valueDomainConformanceRate[0..1] : Real

This data quality measure indicates the number of items in the dataset that are in conformance with their value domain in relation to the total number of items in the dataset. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

valueDomainNonConformanceRate[0..1] : Real

This data quality measure indicates the number of items in the dataset that are not in conformance with their value domain in relation to the total number of items in the dataset. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 5 items that are in conformance and there are 100 of the items in the dataset then the ratio is 5/100 and the reported rate = 0.05.

DQ_ThematicClassificationCorrectness

Comparison of the classes assigned to features or their attributes to a universe of discourse. [Per ISO 19113]

For example, ground truth or reference dataset.

Public Attributes:

numberOfIncorrectlyClassifiedItems[0..1] : Integer

This data quality measure is a count of the number of incorrectly classified features. [Adapted from ISO 19138]

This is an integer count.

miscalculationRate[0..1] : Real

This data quality measure indicates the number of incorrectly classified features in relation to the number of features that are supposed to be there. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 1 items that are classified incorrectly and there are 100 of the items in the dataset then the ratio is 1/100 and the reported rate = 0.01.

misclassificationMatrix[0..1] : Integer Matrix

This data quality measure is a matrix of integer numbers that indicates the number of items of class (i) classified as class (j). The misclassification matrix is a quadratic matrix with n columns and n rows where n denotes the number of classes under consideration. MCM (i,j) = (# items of class (i) classified as class (j). The diagonal elements of the misclassified matrix contain the correctly classified items, and the off diagonal items contain the number of misclassified errors. [Adapted from ISO 19138]

relativeMiscalculationMatrix[0..1] : Real Matrix

This data quality measure is a matrix of real numbers that indicates the number of items of class (i) classified as class (j) divided by the number of items of class (i) * 100 represented as a percentage. The misclassification matrix has n columns and n rows where n denotes the number of classes under consideration. RMCM (i,j) = (# items of class (i) classified as class (j) / number of items of class (i)) *100. [Adapted from ISO 19138]

kappaCoefficient[0..1] : Real

This data quality measure is real number coefficient to quantify the proportion of agreement of assignments to classes by removing misclassifications. [Adapted from ISO 19138]

DQ_TopologicalConsistency

Measures of the topological consistency of geometric representations of features. [Adapted from ISO 19138]

Note: in ISO 19115, this is “Correctness of the explicitly encoded topological characteristics of a dataset”, but ISO 19138 states that the measures “will not serve as measures of the consistency of explicit descriptions of topology using the topological objects specified in ISO 19107”, and S-100 does not explicitly encode geometry.

Public Attributes:

numberOfFaultyPointCurveConnections[0..1] : Integer

This data quality measure is a count of the number of faulty point-curve connections in the dataset. A point curve connection exists where different curves touch. These curves have an intrinsic topological relationship that has to reflect the true constellation. For example, two point-curve connections exist when there should only be one. [Adapted from ISO 19138]

This is an integer count.

rateOfFaultyPointCurveConnections[0..1] : Real

This data quality measure indicates the number of faulty link-node connections in relation to the number of supposed link-node connections. This data quality measure gives the erroneous point-curve connections in relation to the total number of point-curve connections. [Adapted from ISO 19138]

This is a RATE which is a ratio, and is expressed as a REAL number representing the rational fraction corresponding to the numerator and denominator of the ratio.

For example, if there are 2 items that are faulty link-node connections and there are 100 of the connections in the dataset then the ratio is 2/100 and the reported rate = 0.02.

numberOfMissingConnectionsUndershoots[0..1] : Integer

This data quality measure is a count of items in the dataset within the parameter tolerance that are mismatched due to undershoots. [Adapted from ISO 19138]

This is an integer count.

numberOfMissingConnectionsOvershoots[0..1] : Integer

This data quality measure is a count of items in the dataset within the parameter tolerance that are mismatched due to overshoots. [Adapted from ISO 19138]

This is an integer count.

numberOfInvalidSlivers[0..1] : Integer

This data quality measure is a count of all items in the dataset that are invalid sliver surfaces. A sliver is an unintended area that occurs when adjacent surfaces are not digitized properly. The borders of the adjacent surfaces may unintentionally gap or overlap to cause a topological error. [Adapted from ISO 19138]

This is an integer count.

numberOfInvalidSelfIntersects[0..1] : Integer

This data quality measure is a count of all items in the dataset that illegally intersect with themselves. [Adapted from ISO 19138]

This is an integer count.

numberOfInvalidSelfOverlaps[0..1] : Integer

This data quality measure is a count of all items in the dataset that illegally self-overlap. [Adapted from ISO 19138]

This is an integer count.

S-100 – Part 5

Feature Catalogue

Page intentionally left blank

Contents

Page intentionally left blank

5-1 Scope

This Part provides a standard framework for organizing and reporting the classification of real world phenomena in a set of geographic data. It defines the methodology for classification of the feature types and specifies how they are organized in a Feature Catalogue and presented to the users of a set of geographic data. This methodology is applicable to creating catalogues of feature types in previously uncatalogued domains and to revising existing Feature Catalogues to comply with standard practice. It applies to the cataloguing of feature types that are represented in digital form. Its principles can be extended to the cataloguing of other forms of geographic data.

A Feature Catalogue shall be defined for each Product Specification.

This Part is applicable to the definition of geographic features at the type level but not applicable to the representation of individual instances of each type.

5-2 Conformance

This profile conforms to conformance class 2 of ISO 19106:2004. The following is a brief description of the specializations and generalizations where the profile differs from ISO 19110.

1) New abstract classes, *S100_FC_Item*, *S100_FC_NamedType*, and *S100_FC_ObjectType* are introduced.
2) A new class, S100_FC_InformationType is introduced.
3) New classes, S100_FC_FeatureBinding, S100_FC_InformationBinding and S100_FC_AttributeBinding are introduced.
4) A new class, S100_CD_AttributeConstraints is introduced.
5) The class FC_FeatureAttribute is specialized to be the abstract class *S100_FC_Attribute*.
6) New classes, S100_FC_SimpleAttribute and S100_FC_ComplexAttribute are introduced.
7) The classes FC_InheritanceRelation, FC_FeatureOperation, FC_Binding, FC_Constraint and FC_BoundFeatureAttribute are not used.

Further reference or explanation of the above changes can be found in the following text where appropriate.

5-3 Normative References

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

ISO 19110:2005, *Geographic Information – Methodology for feature cataloguing*

5-4 Principal Requirements

5-4.1 Feature Catalogue

An S-100 based Feature Catalogue presents the abstraction of reality represented in one or more sets of geographic data as a defined classification of phenomena. The basic level of classification in the Feature Catalogue is the feature type. Features and attributes are bound in a Feature Catalogue. The definitions of features and attributes are drawn from a Feature Concept Dictionary. A Feature Catalogue shall be available in electronic form (for example XML) for any set of geographic data that contains features. A Feature Catalogue may also comply with the specifications of this component of S-100 independently of any existing set of geographic data.

5-4.2 Information Elements

5-4.2.1 Introduction

The following clauses specify general and specific requirements for Feature Catalogue information elements. A Feature Catalogue generally consists of a list of named types, a list of properties for named types and the information on how both are linked together. Furthermore it contains a list of sources for its definitions. The model is primarily based on the ISO 19110 standard but there are both extensions and differences in this model.

There are two major extensions to the feature types: information types and complex attributes. To achieve a greater flexibility in modelling the data within a data set it is necessary to define complex structures of information. Both extensions allow the creation of those structures. Whereas complex attributes define complex characteristics for one named type, information types can be shared.

Unlike feature types, which are an abstraction of real world phenomena, information types are just shareable structured pieces of information. In a geographic data set they will be associated to feature types or to other information types. Both types: feature and information, have many common characteristics. This is accommodated by deriving both types from a common abstract base class: the named type.

Complex attributes are an aggregation of other attributes which may be either simple or complex.

The arrangement of content may be different depending on format, for example printed document, XML, hypertext etc.

5-4.2.2 Named Types

5-4.2.2.1 Common Characteristics

Feature and information types are inherited (see clause 5-4.2.2.2 below) from the abstract class *S100_FC_NamedType*. This class describes all common characteristics, for example, the name and the definition of the corresponding type. Furthermore a code has to be defined for the type. This code will later be used to identify an instance of a named type in a geographic data set. If the definition is taken from a Feature Concept Dictionary that reference is also given.

Feature and information types can be derived from other feature or information types. This includes the possibility that some types are abstract, meaning no instances of such types can be in a data set. Named types can be characterized by attributes and additional information may be available by information types that are related to them. The former is defined by attribute bindings whereas the latter is achieved by information bindings.

5-4.2.2.2 Inheritance

In data modelling, inheritance is a way to form new types using types that have already been defined. The new types, known as derived types (or sub-types), take over (or inherit) properties of the pre-existing types, which are referred to as base types (or super-types). The derived types may define new additional properties, but also change existing properties, the latter is called overriding. This is used to assign unique property values to sub-types such as name and definition but overriding of characteristics such as bindings to attributes should be avoided by only including common characteristics in the super-type. In the scope of a Feature Catalogue both feature and information types can be derived from other feature or information types. But a feature type cannot be derived from an information type or vice versa. Attributes and associations defined for the super-type will also belong to the sub-type. The definition of the sub-type is usually redefined. In the context of this standard inheritance will be always simple, meaning each type cannot be derived from more than one super-type.

EXAMPLE 1 Cardinal and lateral buoys may be derived from an (abstract) type buoy. The super-type already defines attributes like colour, shape, name, and associations to lights or top marks. The derived types add special information only valid for the specialized type like category of cardinal mark or category of lateral mark respectively.

Inheritance builds hierarchical structures which may become difficult to manage if they are too complex or not sufficiently mature. It is generally good design practice to keep the depth of an inheritance tree as shallow as possible. On the other hand, sometimes inheritance trees simplify models by grouping types which are derived from the same basic concept and which have the same characteristics, so inheritance even at multiple levels should be used where appropriate.

Inheritance relationships between types in a Feature Catalogue generally correspond to inheritance relationships in the Application Schema. Determinations of when to use inheritance and to what degree and level are information modelling questions and should be addressed by Application Schema designers and project teams taking into consideration factors such as Application Schema and Feature Catalogue complexity, maintenance, application requirements, etc.

EXAMPLE 2 In the information model for the ENC Product Specification, all geographic feature types have information bindings to the information type *SupplementaryInformation* and feature bindings to the cartographic feature *TextAssociation*. Defining a common super-type for all geographic features would allow these two bindings to be made to the super-type instead of repeating them in every geographic feature type.

EXAMPLE 3: In an Application Schema for an “Aids to Navigation” Product Specification, classes defining different types of beacons have many of the same attributes. Also, classes defining different types of buoys share the same characteristics. Super-types *GenericBuoy* and *GenericBeacon* are therefore defined. Further, buoys and beacons can all act as structure objects, and there are also other features which can also play the role of structure objects, so another super-type is introduced for generic structure features. *AidsToNavigation*, *StructureObject*, *GenericBuoy*, and *GenericBeacon* are all abstract classes. The Structure/Equipment association is made between the classes Structure and Equipment and applies to all sub-types of these classes, for example any *CardinalBuoy* can fill the parent role in a Structure/Equipment association with any sub-type of Equipment.

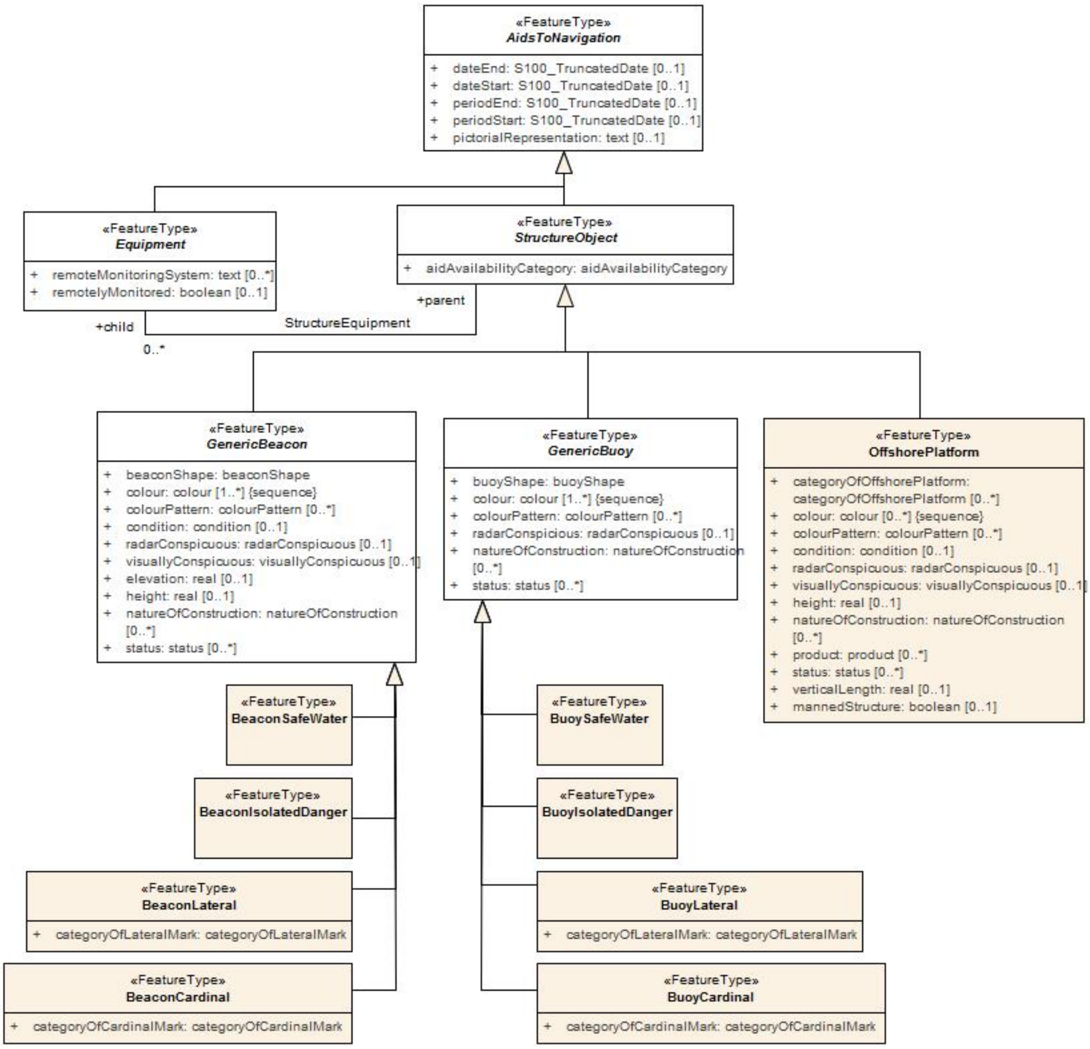


Figure 5-1 – Inheritance Example

5-4.2.2.2.1. Considerations for Product Specifications (Informative)

In general the need for inheritance increases with increasing numbers of concepts which can be grouped under a higher-level concept, or as more characteristics are shared between similar types, or even if several different types share some characteristics.

The advantage of excluding inheritance from Feature Catalogues is mainly structural simplification (and consequently simpler processing) since abstract types and inheritance hierarchies need not be implemented; also in S-100-based Product Specifications, inherited enumerated attributes can have different lists of allowed values for different sub-types. The disadvantages include (probable) increases in the volume of the Feature Catalogue especially if many features or information types have common attributes or associations, and increased complexity for maintenance (an update to an attribute nominally bound to a super-type would have to be made to each sub-type at all levels, and this would have to be checked before the Feature Catalogue is released). Also, inheritance is a common paradigm in object-oriented programming and may not be a significant issue for implementations.

5-4.2.2.3 Feature Types

Feature types are the basic level of classification in the Feature Catalogue. In addition to the common characteristics they define a feature use type to categorize them. Feature types may be associated to other feature types through feature associations. This will be defined by feature bindings which specify the association as well as the role used for the relationship to the other feature type.

5-4.2.2.4 Information Types

Information types are complex pieces of information in a data set that can be shared between many other feature or information types. In regards to their structure, they can also be seen as feature types without a geometric property which have a structure similar to feature types and are categorized as a separate item type.

5-4.2.3 Properties

5-4.2.3.1 Common Characteristics

Properties for feature and information types are attributes and association roles although the latter only applies to feature types. The common characteristics include name, definition, remarks etc. A reference to a Feature Concept Dictionary may be defined.

5-4.2.3.2 Attributes

Attributes carry the characteristics of feature and information types. Unlike information types they cannot be shared between different instances. That is, an instance of an attribute belongs to one and only one feature or information type. In this standard there are two different kinds of attributes: simple and complex. Simple attributes carry the value itself, and complex attributes are aggregations of other attributes to achieve a complex and hierarchical data structure.

5-4.2.3.3 Simple Attributes

Simple attributes are designed to carry a value. In the Feature Catalogue the domain of the value shall be specified. All attribute values are value types. Part 2a-4.2.9 contains the full list of value types and their definitions. If the value type is an enumeration, or a codelist of type “open enumeration”, a list of ‘Listed Values’ will be defined. For codelists of type open or closed dictionary, a URI identifying a “dictionary” (or “vocabulary”) will be provided as a definition.

Furthermore the value domain can be constrained by the following:

1) The length of the text;
2) A format specification for structured text;
3) A numeric range.

Details are in Appendix 5-A.

5-4.2.3.4 Complex Attributes

Complex attributes are aggregations of other attributes that are either simple or complex. The aggregation is defined by means of attribute bindings.

5-4.2.3.5 Association Roles

An association role describes the nature of the relationship from one feature type to another feature type in a feature association. In this standard each association has exactly two roles. Either or both may be a default. The documentation of Application Schemas must specify the rule used for default names. Different rules for default names may apply to different associations in the same Application Schema, but each role shall have an unambiguous name, be it an explicit role name or a default role name.

5-4.2.4 Feature Associations

Feature associations describe the relationships between feature types. Feature associations have a name, definition, remarks, code etc. Each association uses two roles that define the directed use of the relationship. Either or both of the roles may be a default as described in Part 3.

EXAMPLE 1 Structure – Equipment is an example of an association with two roles.

EXAMPLE 2 theAuthority – theContactDetails is an example of an association between classes Authority and ContactDetails which uses two default roles.

5-4.2.5 Bindings

5-4.2.5.1 Attribute Bindings

The following use cases for attribute bindings exist:

1. Defining the attribute for feature types;
2. Defining the attributes for information types;
3. Defining the attributes for feature associations;
4. Defining the attributes for information associations;
5. Defining the aggregation of attributes for a complex attribute.

The binding specifies the target attribute and the Multiplicity of the attribute. The Multiplicity indicates how many instances of an attribute can be used. Bindings are used to define whether an attribute is mandatory (1..n) or optional (0..n). If the Multiplicity allows more than one instance of an attribute a Boolean flag indicates if the sequence of attributes has a meaning.

If the attribute is a simple attribute with a data type of Enumeration, a list of permitted values can be specified. An empty list indicates that all values defined for the attribute in the Feature Catalogue are valid.

5-4.2.5.2 Feature Bindings

The feature binding describes the association between two feature types. Each feature binding is contained within the type definition for a “source” feature type in the Feature Catalogue, and describes the relation of a feature type (the “target”) to the source feature type. A feature binding specifies:

- the name of the feature association;
- the target feature type;
- the role of the target feature type in relation to the source feature (the “role” is the name of the association end at the target);
- the type of association end at the target (ordinary association, aggregation, or composition); and
- the multiplicity of the target feature type.

EXAMPLE: The **TrafficSeparationScheme** feature type is associated to the **TrafficSeparationSchemeLanePart** feature by the ***TrafficSeparationSchemeAggregation*** association. This association is an aggregation and is depicted in the Figure 5-2 UML diagram below:

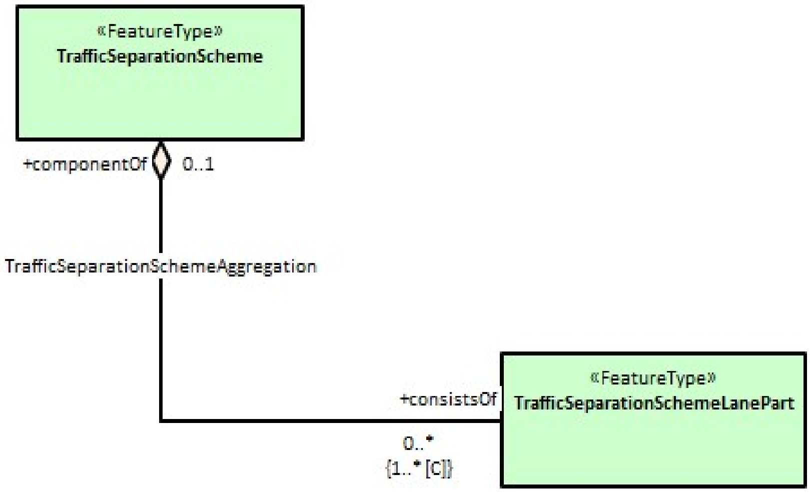


Figure 5-2 – TrafficSeparationSchemeAggregation association between TrafficSeparationScheme and TrafficSeparationSchemeLanePart feature classes

In accordance with UML conventions, the diamond at the TrafficSeparationScheme end means that TrafficSeparationScheme is the “whole” or “container” in the association and TrafficSeparationSchemeLanePart is the “part” or “containee”. The feature bindings in the respective feature types in the XML Feature Catalogue are:

In the TrafficSeparationScheme:

```
<S100FC:featureBinding roleType="association">
    <S100FC:multiplicity>
        <S100Base:lower>0</S100Base:lower>
        <S100Base:upper xsi:nil="true" infinite="true"/>
    </S100FC:multiplicity>
    <S100FC:association ref="TrafficSeparationSchemeAggregation"/>
    <S100FC:role ref="consistsOf"/>
    <S100FC:featureType ref="TrafficSeparationSchemeLanePart"/>
</S100FC:featureBinding>
```

In TrafficSeparationSchemeLanePart:

```
<S100FC:featureBinding roleType="aggregation">
    <S100FC:multiplicity>
        <S100Base:lower>0</S100Base:lower>
        <S100Base:upper xsi:nil="false" infinite="false">1</S100Base:upper>
    </S100FC:multiplicity>
    <S100FC:association ref="TrafficSeparationSchemeAggregation"/>
    <S100FC:role ref="componentOf"/>
    <S100FC:featureType ref="TrafficSeparationScheme"/>
</S100FC:featureBinding>
```

Note that data formats may impose constraints on whether bindings are actually encoded in either of the participating feature instances in datasets.

5-4.2.5.3 Information Bindings

The information binding describes the association between a feature and information type or between two information types. Each information binding is contained within the type definition for a “source” feature or information type in the Feature Catalogue, and describes the relation of an information type (the “target”) to the source type. An information binding specifies:

- the name of the information association;
- the target information type;
- the role of the target information type in relation to the source feature or information type (the “role” is the name of the association end at the target);
- the type of association end at the target (ordinary association, aggregation, or composition); and
- the multiplicity of the target information type.

The structure in the XML Feature Catalogue is similar to the example in clause 5-4.2.5.2 except that one or both of the types will be an information type and the XML will be for “informationBinding” instead of “featureBinding”.

As for feature bindings, data formats may impose constraints on whether bindings are actually encoded in either of the participating feature instances in datasets (for example, that for an information association linking a feature to an information type, the binding is encoded only in the feature instance and therefore the Feature Catalogue may not include the binding in the information type, only in the feature type).

5-4.2.6 Definitions and source references

5-4.2.6.1 Definition sources

This is a list of source documents for the definitions used in the Feature Catalogue. They are given with their citation information. Usually the definitions will come from a Feature Concept Dictionary but other sources are possible. It is also valid that a definition originates from the Feature Catalogue; in this case there will be no reference to a definition source.

5-4.2.6.2 Definition references

This information carries the link to the definition source. It points to a definition source and defines the place in that source by means of an identifier. In cases where the source is a Feature Concept Dictionary maintained as a Register this reference will be the item identifier.

5-4.2.7 Completeness

A template for the representation of feature classification information is specified in the following model (Appendix 5-A (normative), Figure 5-A.1). A Feature Catalogue prepared according to this template shall document all of the feature types and information types found in a given set of geographic data. The Feature Catalogue shall include identification information as specified. The Feature Catalogue shall include definitions and descriptions of all feature and information types contained in the data, including any feature attributes and feature associations contained in the data that are associated with each feature type. To ensure predictability and comparability of Feature Catalogue content across different applications, it is recommended that the Feature Catalogue should include only the elements specified in the tables shown at Appendix 5-A (normative) below.

Appendix 5-A
Feature Catalogue Model
(normative)

This appendix presents the S-100 Feature Catalogue. Figure 5-A-1 is the S-100 Feature Catalogue modelled in UML and Tables 5-A-1 to 5-A-20 illustrate the structure of the Feature Catalogue in conformance to the model shown.

Page intentionally left blank

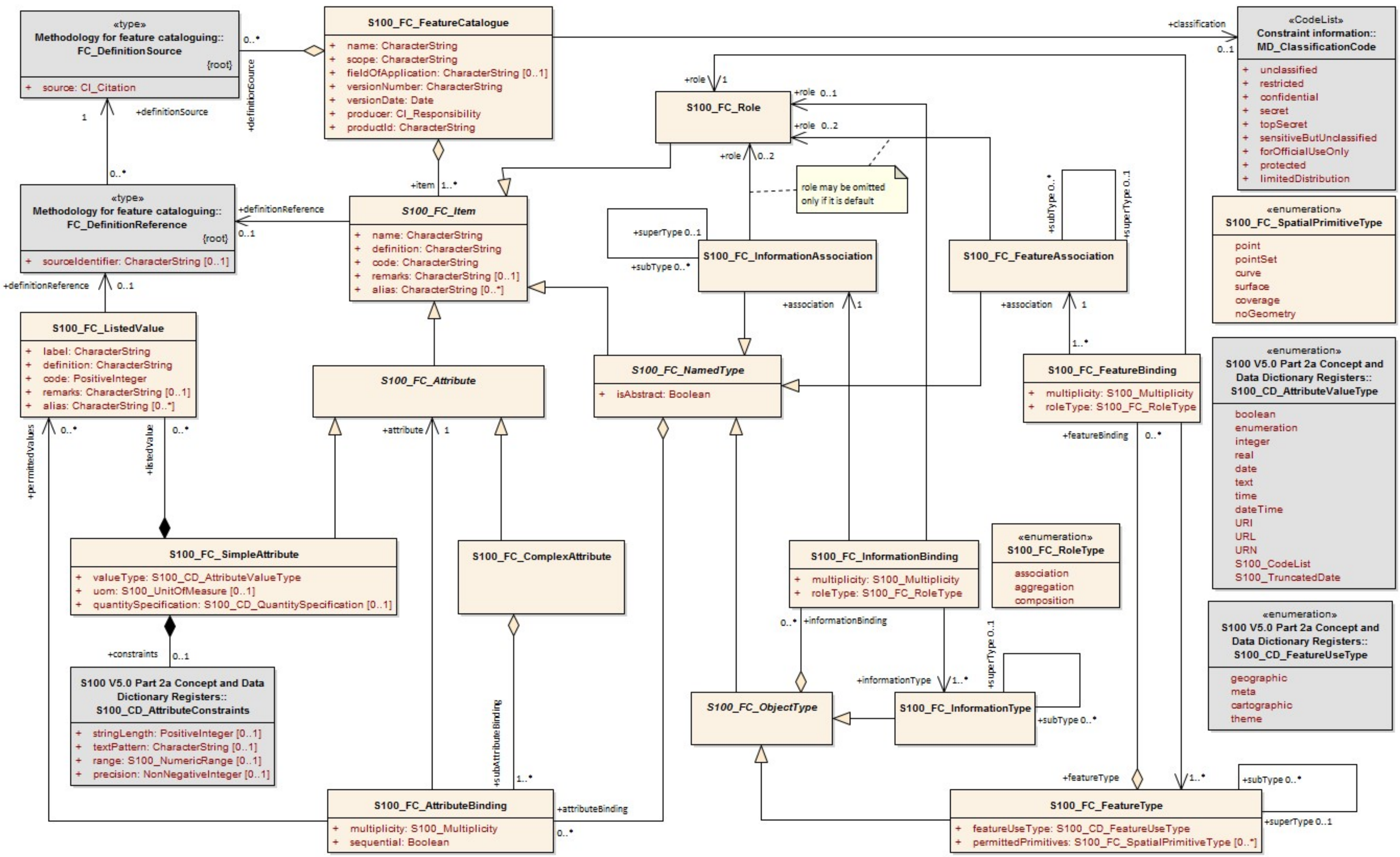


Figure 5-A-1. Feature Catalogue — UML Model

Table 5-A-1 — S100_FC_FeatureCatalogue

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_FeatureCatalogue	A Feature Catalogue contains its identification and contact information, and definition of some number of feature types with other information necessary for those definitions	-	-	-
Attribute	name	Name for this Feature Catalogue	1	CharacterString	
Attribute	scope	Subject domain of feature types defined in this Feature Catalogue	1	CharacterString	
Attribute	fieldOfApplication	Description of the kind of use to which this Feature Catalogue may be put	0..1	CharacterString	
Attribute	versionNumber	Version number of this Feature Catalogue, which may include both a major version number or letter and a sequence of minor release numbers or letters, such as “3.2.4a.” The format of this attribute may differ between cataloguing authorities	1	CharacterString	
Attribute	versionDate	Effective date of this Feature Catalogue	1	Date	
Attribute	productId	The ID of the product for which the Catalogue is intended	1	CharacterString	
Attribute	producer	Name, address, country, and telecommunications address of person or organization having primary responsibility for the intellectual content of this Feature Catalogue	1	CI_Responsibility	CI_Responsibility>CI_Individual or CI_Responsibility>CI_Organisation
Role	item	List of items defined by this Feature Catalogue; items are feature types, information types, feature associations, information associations, attributes, and roles	1..*	S100_FC_Item	Aggregation
Role	definitionSource	List of sources of definitions of items and listed values that are defined by this Feature Catalogue. Usually those sources are Feature Data Dictionaries	0..*	FC_DefinitionSource	Aggregation

Role	classification	The classification of the Feature Catalogue.	0..1	MD_ClassificationCode	1 unclassified 2 restricted 3 confidential 4 secret 5 top secret 6 sensitive but unclassified 7 for official use only 8 protected 9 limited distribution

Table 5-A-2 — FC_DefinitionSource

Role Name	Name	Description	Mult	Type	Remarks
Class	FC_DefinitionSource	Class that specifies the source of a definition	-	-	-
Attribute	source	Actual citation of the source, sufficient to identify the document and how to obtain it	1	CI_Citation	

Table 5-A-3 — FC_DefinitionReference

Role Name	Name	Description	Mult	Type	Remarks
Class	FC_DefinitionReference	Class that links a data instance to the source of its definition	-	-	-
Attribute	sourceIdentifier	Information to locate the definition in the source document. The format of this information is specific to the structure of the source document	1	CharacterString	Includes online dictionaries or “vocabularies” used by dictionary-type codelist attributes
Role	definitionSource	The source of the definition	1	FC_DefinitionSource	

Table 5-A-4 — S100_FC_Item

Role Name	Name	Description	Mult	Type	Remarks
Class	*S100_FC_Item*	Abstract base class that defines the common properties of all items in the Feature Catalogue; items are feature types, information types, feature associations, information associations, attributes and roles	-	-	Abstract class

Attribute	name	Name of the item	1	CharacterString	
Attribute	definition	Definition of the named type in a natural language	1	CharacterString	
Attribute	code	Code that uniquely identifies the named type within the Feature Catalogue	1	CharacterString	
Attribute	remarks	Further explanation about the item	0..1	CharacterString	
Attribute	alias	Equivalent name(s) of this item	0..*	CharacterString	
Role	definitionReference	The link to the source of the definition	0..1	FC_DefinitionReference	

Table 5-A-5 — S100_FC_NamedType

Role Name	Name	Description	Mult	Type	Remarks
Class	*S100_FC_NamedType*	Abstract base class that defines the common properties for feature types and information types	-	-	Abstract class
Attribute	isAbstract	Indicates if instances of this named type can exist in a geographic data set. Abstract types cannot be instantiated but serve as base classes for other (non-abstract) types	1	Boolean	
Role	attributeBinding	List of bindings to attributes which describe the characteristic of this named type	0..*	S100_FC_AttributeBinding	Aggregation

Table 5-A-6 — S100_FC_ObjectType

Role Name	Name	Description	Mult	Type	Remarks
Class	*S100_FC_ObjectType*	Abstract base class that defines the common properties for feature types and information types	-	-	Abstract class; derived from S100_FC_NamedType
Role	informationBinding	List of bindings to information types that can be associated to this object type by means of an information association	0..*	S100_FC_InformationBinding	Aggregation

Table 5-A-7 — S100_FC_InformationType

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_InformationType	Class that defines all properties of an information type	-	-	Derived from S100_FC_NamedType
Role	superType	Indicates the information type from which an information type is derived. The sub-type will inherit all properties from its super-type: Name, definition and code will usually be overridden by the sub-type, although new properties may be added to the sub-type	0..1	S100_FC_InformationType	
Role	subType	Indicates the information types which are derived from an information type	0..*	S100_FC_InformationType	

Table 5-A-8 — S100_FC_FeatureType

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_FeatureType	Class that defines all properties of a feature type	-	-	Derived from S100_FC_NamedType
Attribute	featureUseType	The use type of this feature type	1	S100_CD_FeatureUseType	
Attribute	permittedPrimitives	The combination of 0 or more spatial primitives permitted for feature type	0..*	S100_FC_SpatialPrimitiveType	
Role	featureBinding	List of bindings to feature types that can be related to this feature type by means of a feature association	0..*	S100_FC_FeatureBinding	Aggregation
Role	superType	Indicates the feature type from which a feature type is derived. The sub-type will inherit all properties from its super-type: Name, definition and code will usually be overridden by the sub-type, although new properties may be added to the sub-type If permittedPrimitives is present in a sub-type it overrides permittedPrimitives in any of its super-types	0..1	S100_FC_FeatureType	
Role	subType	Indicates the feature types which are derived from a feature type	0..*	S100_FC_ FeatureType	

Example: If a super-type allows point and area primitives and the sub-type only curve primitives, instances of the sub-type must indicate location with curve spatial objects. Sub-types of the sub-type will take only line primitives unless they specify their own permitted primitives.

Table 5-A-9 — S100_FC_InformationAssociation

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_InformationAssociation	An information association describes the relationship between an object (feature or information type) and an information type	-	-	Derived from S100_FC_NamedType Individual Product Specifications may restrict directionality
Role	role	The role of the association	0..2	S100_FC_Role	Default role name if missing Product Specification can constrain further
Role	superType	Indicates the information association from which an information association is derived. The sub-type will inherit all properties from its super-type: Name, definition and code will usually be overridden by the sub-type, although new properties may be added to the sub-type	0..1	S100_FC_InformationAssociation	
Role	subType	Indicates the information associations which are derived from an information association	0..*	S100_FC_InformationAssociation	

Table 5-A-10 — S100_FC_FeatureAssociation

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_FeatureAssociation	A feature association describes the relationship between two feature types A feature association is bidirectional and has a separate role for each direction	-	-	
Role	role	The role of the association	0..2	S100_FC_Role	
Role	superType	Indicates the feature association from which a feature association is derived. The sub-type will inherit all properties from its super-type: Name, definition and code will usually be overridden by the sub-type, although new properties may be added to the sub-type	0..1	S100_FC_FeatureAssociation	
Role	subType	Indicates the feature associations which are derived from a feature association.	0..	S100_FC_FeatureAssociation	

Table 5-A-11 — S100_FC_Role

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_Role	A role which can be used in a feature association or an information association	-	-	Derived from S100_FC_Item

Table 5-A-12 — S100_FC_Attribute

Role Name	Name	Description	Mult	Type	Remarks
Class	*S100_FC_Attribute*	Abstract base class for the two kinds of attributes: simple attributes and complex attributes. Attributes carry the characteristics of named types	-	-	Abstract derived from S100_FC_Item

Table 5-A-13 — S100_FC_SimpleAttribute

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_SimpleAttribute	Attribute that carries a value	-	-	Derived from S100_FC_Attribute
Attribute	valueType	The value type of this feature attribute	1	S100_CD_AttributeValueType	
Attribute	uom	Unit of measure used for values of this feature attribute	0..1	S100_UnitOfMeasure	
Attribute	quantitySpecification	Specification of the quantity	0..1	S100_CD_QuantitySpecification	
Role	constraints	Constraints which may apply to the attribute	0..1	S100_CD_AttributeConstraints	Composition
Role	listedValue	Set of listed values for an enumerated attribute domain	0..*	S100_FC_ListedValue	Composition. Applies only if valueType is Enumeration or S100_Codelist (with codelistType=open enumeration)

Table 5-A-14 — S100_FC_ComplexAttribute

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_ComplexAttribute	A complex attribute consists of a list of sub-attributes which can be both simple and complex attributes	-	-	Derived from S100_FC_Attribute
Role	subAttributeBinding	List of attribute bindings to the sub-attributes	1..*	S100_FC_AttributeBinding	Aggregation

Table 5-A-15 — S100_FC_ListedValue

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_ListedValue	Value of an enumerated attribute domain, including its codes and definition	-	-	
Attribute	label	Descriptive label that uniquely identifies one value of the feature attribute	1	CharacterString	
Attribute	definition	Definition of the listed value in a natural language	1	CharacterString	
Attribute	code	Numeric code that uniquely identifies the listed value for the corresponding feature attribute	1	PositiveInteger	
Attribute	remarks	Further explanation about the listed value	0..1	CharacterString	
Attribute	alias	Equivalent name(s) of this listed value	0..*	CharacterString	
Role	definitionReference	The link to the source of the definition	0..1	FC_DefinitionReference	

Table 5-A-16— S100_FC_AttributeBinding

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_AttributeBinding	Class that is used to describe the specifics of how an attribute is bound to a particular named type or a complex attribute	-	-	
Attribute	multiplicity	Multiplicity defining how many instances of the attribute can be part of the named type or complex attribute	1	S100_Multiplicity	
Attribute	sequential	Describes if the sequence of the attributes is meaningful or not	1	Boolean	Applies only to attributes which may occur more than once
Role	permittedValues	Permissible values of the attribute	0..*	S100_FC_ListedValue	Applies only to attributes of data type enumeration
Role	attribute	The attribute that is bound to the item or complex attribute	1	S100_FC_Attribute	

Table 5-A-17 — S100_FC_InformationBinding

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_InformationBinding	Class describing the use of an information type by a named type	-	-	
Attribute	multiplicity	Multiplicity defining how many instances of the target information type can be linked to one instance of the named type	1	S100_Multiplicity	
Attribute	roleType	The nature of the association end	1	S100_FC_RoleType	
Role	role	The role used for the binding. It must be part of the association used for the binding and defines the end of the association	0..1	S100_FC_Role	
Role	association	The association used for the binding; defining also the role	1	S100_FC_InformationAssociation	
Role	informationType	The target information type	1..*	S100_FC_InformationType	

Table 5-A-18 — S100_FC_FeatureBinding

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_FC_FeatureBinding	Class describing the relationship from one feature type to another feature type by means of a feature association	-	-	
Attribute	multiplicity	Multiplicity defining how many instances of the target feature type can be linked to one instance of the source feature type	1	S100_Multiplicity	
Attribute	roleType	The nature of the association end	1	S100_FC_RoleType	
Role	featureType	The target feature type	1..*	S100_FC_FeatureType	
Role	role	The role used for the binding. It must be part of the association used for the binding and defines the end of the association	1	S100_FC_Role	
Role	association	The association used for the binding	1	S100_FC_FeatureAssociation	

Table 5-A-19 — S100_FC_RoleType

Item	Name	Description	Remarks
Enumeration	S100_FC_RoleType	Defines the type of an association end (that is, a “role”)	
Literal	association	The association end is an ordinary linkage. (In UML terms, the role type is “aggregationKind=ordinary” and the link in a diagram does not have a diamond)	The object at this end may be participating in an ordinary association, an aggregation, or a composition
Literal	aggregation	The association end is a UML aggregation. (In UML terms, the role type is “aggregationKind=aggregation” and the link in a diagram has an unfilled diamond at this association end)	The object at this end is the “owner”, “whole” or “container” in an aggregation association
Literal	composition	The association end is a UML aggregation. (In UML terms, the role type is “aggregationKind=composition” and the link in a diagram has a filled diamond at this association end)	The object at this end is the “owner”, “whole” or “container” in an composition association

NOTE: If one end of the association is “aggregation” or “composition”, the other end must be coded as “association”.

Table 5-A-20 — S100_FC_SpatialPrimitiveType

Item	Name	Description	Remarks
Enumeration	S100_FC_SpatialPrimitiveType	Specifies spatial primitives permitted for use with a feature instance	
Literal	point	Point spatial primitive	**GM_Point**
Literal	pointSet	Point set spatial primitive	**GM_MultiPoint**
Literal	curve	Curve spatial primitive	**GM_OrientableCurve**
Literal	surface	Surface spatial primitive	**GM_OrientableSurface**
Literal	coverage	Coverage spatial primitive	**CV_Coverage**
Literal	noGeometry	The feature type is not associated with a spatial primitive for the location of instances	In some cases, an explicit statement is needed to indicate that there are no spatial primitives for the locations of instances. See the rules for subtypes and super-types in S100_FC_FeatureType

S-100 – Part 6

Coordinate Reference Systems

Page intentionally left blank

Contents

Page intentionally left blank

6-1 Scope

The S-100 standard has been designed for the producers and users of hydrographic information, however its principles can be extended to many other forms of geographic information including maps, and text documents.

The location of an object in the S-100 standard is defined by means of coordinates. Those coordinates relate a feature to a position. This Part describes all elements that are necessary to fully define the referencing by means of coordinate systems and datums. It defines the Conceptual Schema for the description of spatial referencing by coordinates and describes the minimum data required to define 1-, 2- and 3-dimensional spatial coordinate references.

In addition to the elements necessary to define a coordinate reference system this Part also describes operations to transform coordinates from one coordinate reference system to another. This includes operations for datum transformation and map projections.

Coordinate reference systems, as well as single elements to define them, may be registered in a register or defined by an organization in a document. This Part describes how those elements are identified.

A coordinate reference system shall not change with time within the scope of this Part.

6-2 Normative References

The following referenced documents are indispensable for the application of this document. For dated references, only the cited edition applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 19111:2007, *Geographic information — Spatial referencing by coordinates*

ISO/TS 19103, *Geographic information — Conceptual schema language*

ISO 19115, *Geographic information — Metadata*

6-3 Package Overview

6-3.1 The package diagram

Figure 6.1 shows the packages used in this Part and their dependencies.

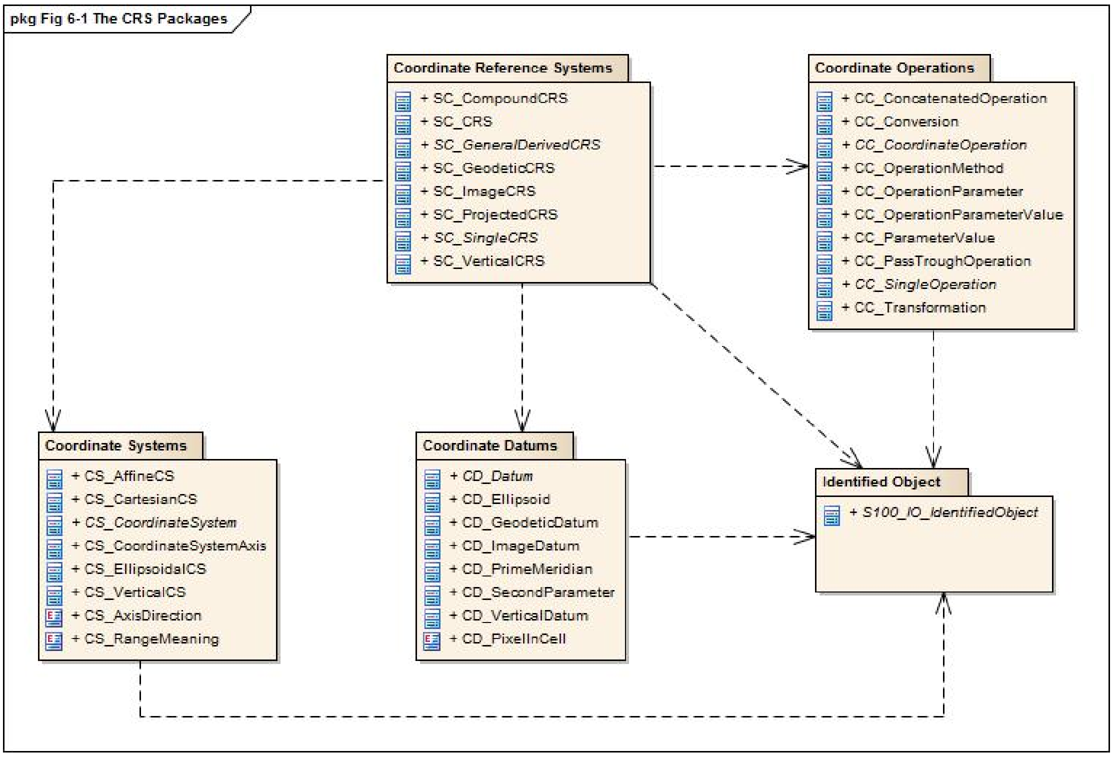


Figure 6-1 — The CRS packages

The elements for referencing spatial objects by use of coordinates are described in five packages. All packages depend on the package “Identified Objects”, which describes the mechanism of linking elements to external definitions.

It also assures that each element can be uniquely named to identify it in a data set or software application. To facilitate the work with class names, every package shall use a prefix for its classes and data types. Table 6-1 below shows the prefixes for various packages:

Table 6-1 — Package prefixes

Package name	Prefix
Identified Objects	IO
Coordinate Reference Systems	SC
Coordinate Systems	CS
Datums	CD
Coordinate Operations	CC

6-4 Package details

6-4.1 The Identified Object package

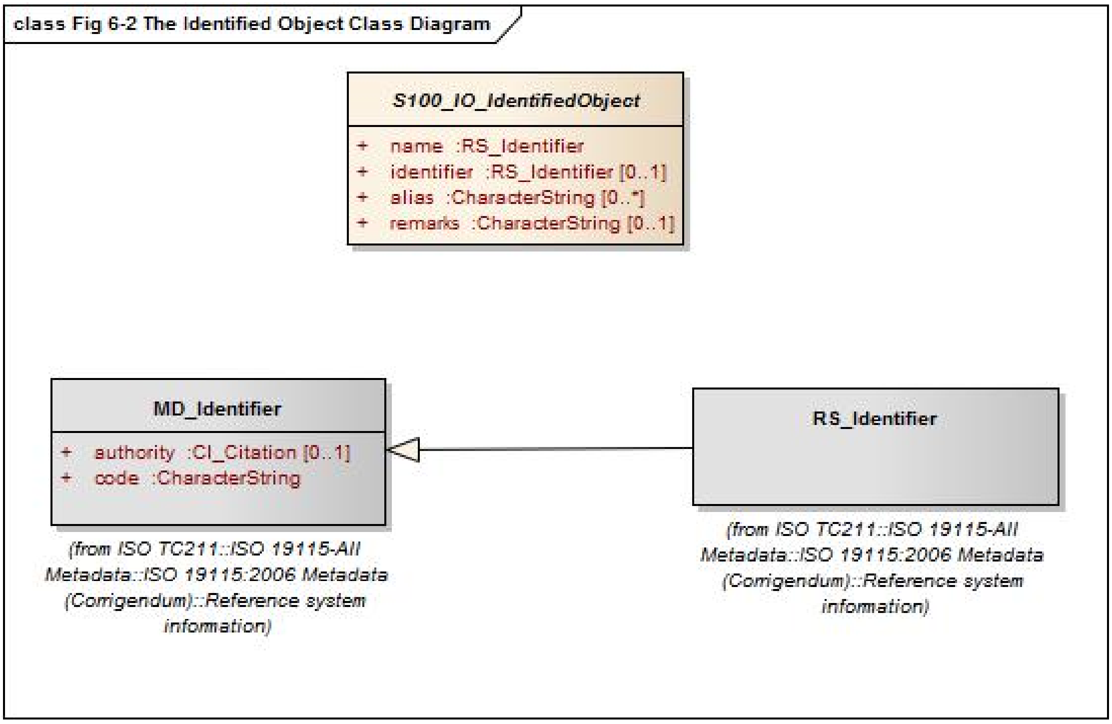


Figure 6-2 — The Identified Object class diagram

NOTE If a class diagram of a package shows classes or types of another package it will be shown with a grey background. In this case not all details of this class are shown; the full details are described in the class diagram of the package where the class belongs.

6-4.1.1 S100_IO_IdentifiedObject

Each class in this part is intended to have a mechanism to be identifiable and/or to identify an external source that is derived from the class S100_IO_IdentifiedObject.

Different from ISO 19111 this class is not derived from an external document but uses members defined by external standards. In addition, no other class in this part is derived from external standards. Where in ISO 19111 those classes inherit essential members, those members will be introduced here in the appropriate package diagram. This should improve the readability of this component and also avoids multiple inheritance where it is not absolutely needed.

6-4.1.2 Class details

Table 6-2 — Properties of the class IO_IdentifiedObject

Name	Type	Card.	Description
name	RS_Identifier	1	The primary name by which the object can be identified
identifier	RS_Identifier	0..1	An identifier that references the (external) definition of the object
alias	GenericName	0..*	An alternative name of the object
remarks	CharacterString	0..1	Comments on or information about the object

The type RS_Identifier (from ISO 19115) has three parts:

1) authority : CI_Citation [0..1];
2) code : CharacterString;
3) codeSpace: CharacterString [0..1]version: CharacterString [0..1].

The type CI_Citation is also defined in ISO 19115, for more details refer there.

6-4.2 The Coordinate Reference Systems package

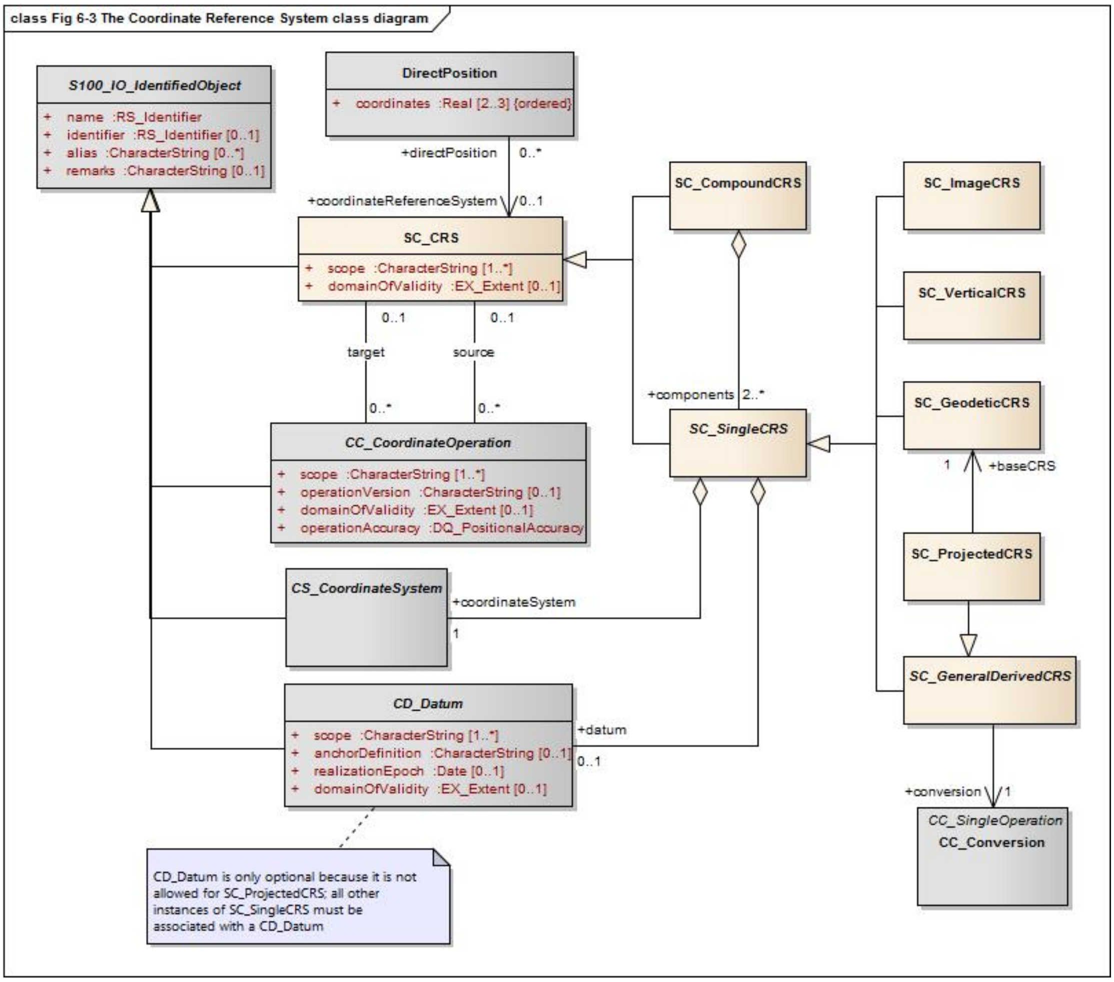


Figure 6-3 — The Coordinate Reference Systems class diagram

This package describes the base class used for all coordinate reference systems and all derived subclasses supported by this component. The diagram also shows the relation to classes in other packages.

A coordinate reference system is a coordinate system that is related to the real world by a datum. Generally, the real world will be the Earth although the principles are not restricted to the Earth. A coordinate reference system (CRS) is either a single CRS or a compound CRS.

6-4.3 Single CRS

A single CRS is defined by a coordinate system and an associated datum. The following types of single CRSs are supported by S-100:

1) Geodetic CRS;
2) Projected CRS;
3) Vertical CRS;
4) ImageCRS.

A geodetic CRS is associated with a geodetic datum. It usually uses an ellipsoidal coordinate system (geodetic latitude, geodetic longitude and ellipsoidal height if 3D). A geodetic CRS can also use a Cartesian coordinate system (3D, fixed to the earth). Coordinates referenced to a Cartesian system are rarely used in data sets but are used as intermediate coordinates during certain coordinate transformations.

A projected CRS is a derived CRS with a geodetic CRS as its base and using a map projection for coordinate conversion. The underlying coordinate system is always a Cartesian coordinate system. Projected CRS are frequently used for national coordinate systems.

A vertical CRS is a 1D CRS for reporting depth or heights and associated with a vertical datum. The ellipsoidal height cannot be captured with a vertical CRS. Ellipsoidal heights are an integral part of a 3D coordinate tuple of a geodetic CRS and cannot exist independently.

An image CRS is associated with an image datum that describes how the image coordinate system is related to the image. This relation is independent of whether or not the image is georeferenced. Georeferencing is performed through a transformation of the image CRS to a geodetic or a projected CRS.

6-4.3.1 Compound CRS

A compound CRS is a combination of two or more single CRSs although the use of more than two components is very unlikely. The components of a compound CRS must be independent. Two CRSs are independent if coordinates conforming to them cannot be changed from one CRS to the other by some coordinate operation. A horizontal and vertical CRS, for example, are independent while two vertical CRSs are not. Nesting of compound CRSs is not permitted, meaning all components must be single CRSs.

Each position in a data set, given with the class DirectPosition, must be bound to a CRS. If in a data set different vertical datums are used for each, a vertical coordinate system has to be defined. Those vertical coordinate systems can then be used as a component in a compound coordinate system to describe a three-dimensional coordinate.

If a data Product Specification allows a choice of geodetic datums, even if only one is allowed in a given dataset, transformation methods must be specified that enable the datasets to be used together in an application.

6-4.3.2 Class details

Table 6-3 — Properties of the class SC_CRS

Name	Type	Card.	Description
scope	CharacterString	1..*	Description of usage, or limitations of usage, for which this CRS is valid
domainOfValidity	EX_Extend	0..1	Area or region in which this CRS is valid

Table 6-4 — Properties of the class SC_SingleCRS

Name	Type	Card.	Description
datum	CD_Datum	0..1	The datum with which the CRS is associated. The datum must be of an appropriate type (vertical or horizontal) for the CRS. Mandatory except for projected CRS, for which it must not be specified – the projected CRS uses the datum of its base CRS
coordinateSystem	CS_CoordinateSystem	1	Coordinate system used by the CRS

Table 6-5 — Properties of the class SC_GeneralDerivedCRS

Name	Type	Card.	Description
conversion	CC_Operation	1	The coordinate conversion method to convert the coordinates from the base to the derived CRS (for example a map projection)

Table 6-6 — Properties of the class SC_Projected CRS

Name	Type	Card.	Description
baseCRS	SC_GeodeticCRS	1	The geodetic CRS on which the CRS is based. In particular the datum of the base CRS is also used for the derived CRS

6-4.4 The Coordinate System package

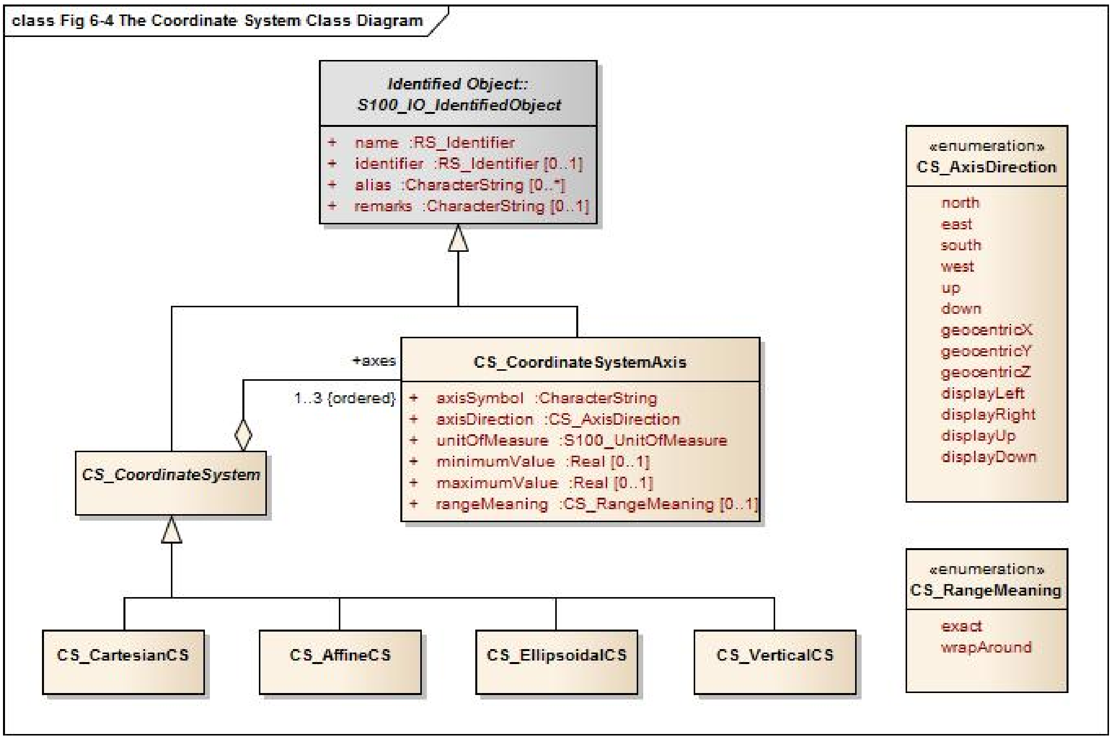


Figure 6-4 — The "Coordinate System" class diagram

A coordinate system comprises a non-repeating, ordered sequence of coordinate axes. The number of axes shall be equal to the number of dimensions of the space which geometry the CRS describes. The order of the coordinate axes is identical to the order of the coordinates in each coordinate tuple described by a CRS using this coordinate system.

This component defines four types of coordinate systems:

1) Cartesian coordinate system;
2) Affine coordinate system;
3) Ellipsoidal coordinate system;
4) Vertical coordinate system.

Each axis is defined by the direction, the value range and the unit of measure used.

A Cartesian coordinate system is a two- or three-dimensional coordinate system with orthogonal straight axes. All axes shall have the same length unit.

An affine coordinate system is a two- or three-dimensional coordinate system with straight axes that are not necessarily orthogonal. All axes shall have the same length unit.

An ellipsoidal coordinate system is a two- or three-dimensional coordinate system which describes coordinates on or nearby the surface of an ellipsoid. The coordinates are: geodetic latitude, geodetic longitude and (in the three-dimensional case) ellipsoidal height.

The geodetic latitude is the angle from the equatorial plane to the perpendicular to the ellipsoid through a given point, northwards treated as positive.

The geodetic longitude is the angle from the prime meridian plane to the meridian plane of a given point, eastward treated as positive.

The ellipsoidal height is the distance of a point from the ellipsoid measured along the perpendicular from the ellipsoid to this point, positive if upwards or outside of the ellipsoid.

A vertical coordinate system is a one-dimensional coordinate system used to record the heights or depths of points. Such a coordinate system is usually dependent on the Earth's gravity field. The following table specifies the type of CRS's that can use the specific type of coordinate system.

Table 6-7 — Coordinate systems used for different CRS's

Coordinate Reference System	Coordinate System	Dimension
Geodetic CRS	Ellipsoidal coordinate system Cartesian coordinate system	2, 3 3
Projected CRS	Cartesian coordinate system	2
Vertical CRS	Vertical coordinate system	1
Image CRS	Cartesian coordinate system Affine coordinate system	2 2

6-4.4.1 Class details

Table 6-8 — Properties of the class CS_CoordinateSystem

Name	Type	Card.	Description
axes	CS_CoordinateSystemAxis	1..3	The axes of the coordinate system. The order is the same as the order of the coordinates in the corresponding positions. The number equals the dimension of the space for which the coordinate system describes the geometry

Table 6-9 — Properties of the class CS_CoordinateSystemAxis

Name	Type	Card.	Description
axisSymbol	CharacterString	1	Abbreviation used for this coordinate system axis.
axisDirection	CS_AxisDirection	1	Direction of the coordinate system axis. For an Earth-fixed coordinate system the value is often approximate and intended to provide a human interpretable meaning to the axis
minimumValue	double	0..1	The minimum value allowed for this axis in the axis' units of measure
maximumValue	double	0..1	The maximum value allowed for this axis in the axis' units of measure
rangeMeaning	CS_RangeMeaning	0..1	The meaning of the value range.
unit of measure	S100UnitOfMeasure	1	The unit of measure for this axis

Table 6-10— Definitions of the enumeration type CS_AxisOrentation

Name	Description
north	Axis positive direction is north. In a geodetic or projected CRS, north is defined through the geodetic datum
east	Axis positive direction is 90° (π/2 radians) clockwise from north
south	Axis positive direction is 180° (π radians) clockwise from north
west	Axis positive direction is 270° /3π/2 radians) clockwise from north
up	Axis positive direction is up relative to gravity
down	Axis positive direction is down relative to gravity
geocentricX	Axis positive direction is in the equatorial plane from the centre of the modelled earth towards the intersection of the equator with the prime meridian
geocentricY	Axis positive direction is in the equatorial plane from the centre of the modelled earth towards the intersection of the equator and the meridian π/2 radians eastwards from the prime meridian
geocentricZ	Axis positive direction is from the centre of the modelled earth parallel to its rotation axis and towards its north pole
displayLeft	Axis positive direction is left in display
displayRight	Axis positive direction is right in display
displayUp	Axis positive direction is up in display
displayDown	Axis positive direction is down in display

Table 6-11 — Definitions of the enumeration type CS_RangeMeaning

Name	Description
exact	Any value between and including minValue and maxValue is valid
wrapAround	The axis is continuous with values wrapping around at the minValue and maxValue. Values with the same meaning repeat modulo (maxValue – minValue). An example for this is the geodetic longitude; the axis is defined as a circle and the values wrap around ±π (±180°)

6-4.5 The Datum package

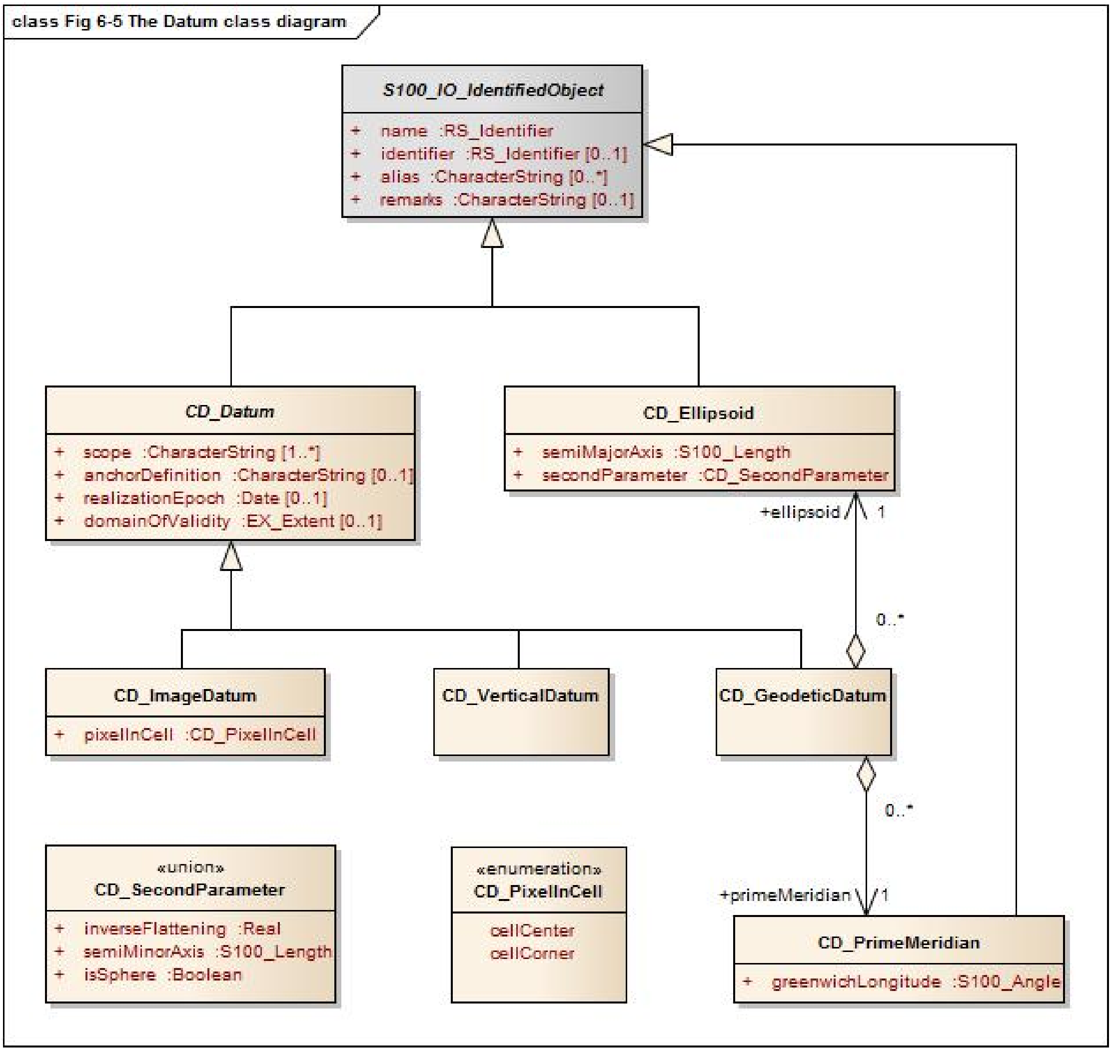


Figure 6-5 — The Datum class diagram

A datum is a parameter or set of parameters that defines the position of the origin, the scale, and the orientation of a coordinate system. Three types of datums are described by S-100:

1) A geodetic datum;
2) A vertical datum;
3) An image datum.

A geodetic datum fixes the relationship of a two- or three-dimensional coordinate system to the Earth. This is done by means of an ellipsoid as the model of the Earth and of a prime meridian as the point of origin of geodetic longitude.

A vertical datum fixes the relationship between gravity-related heights or depths to the Earth. It is used to reference a vertical coordinate system. This relationship may be quite complex.

Ellipsoidal heights are treated as related to a three-dimensional ellipsoidal coordinate system referenced to a geodetic datum. They cannot be referenced by a vertical datum.

An image datum fixes the relationship between a coordinate system and an image. This is independent of whether the image is geo-referenced or not. An image CS is for locating a position within the image, not the position of the object in the real world

An ellipsoid in general is a quadratic surface given in Cartesian coordinates by:

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}=1$$

Where a, b, c are called semi-axes of the ellipsoid.

In the context of geodesy two semi-axes are equal (a=b) and a > c. This figure is also called an oblate spheroid. In S-100 the term ellipsoid is used for this special case and the two semi axes are denoted semi-major axis (a) and semi-minor axis (b), with a > b.

An ellipsoid can be defined either by its two semi-axes or alternatively by its semi-major axis and the inverse flattening: $f^{-1}=\frac{a}{a-b}$

If both semi-axes are equal the ellipsoid is a sphere. In this case the inverse flattening is not defined. (The flattening is 0).

To define the origin on the (circular) axis for the geodetic longitude the prime meridian is used. It is the meridian from which the longitudes of other meridians are quantified.

6-4.5.1 Class details

Table 6-12 — Properties of the class CD_Datum

Name	Type	Card.	Description
scope	CharacterString	1..*	Description of usage, or limitations of usage, for which this datum is valid
anchorDefinition	CharacterString	0..1	A description, possibly including coordinates of an identified point or points, of the relationship used to anchor the coordinate system to the Earth or alternate object For a geodetic datum this is known as the fundamental point For an image datum it is usually a corner of the image or its centre
realizationEpoch	Date	0..1	The time after which this datum definition is valid
domainOfValidity	EX_Extent	0..1	Area or region in which this datum is valid

Table 6-13— Properties of the class CD_Ellipsoid

Name	Type	Card.	Description
semiMajorAxis	Length	1	The length of the semi-major axis of the ellipsoid
secondParameter	CD_SecondParameter	1	The second parameter to define the ellipsoid, either the length of the semi-minor axis or the inverse flattening of the ellipsoid

Table 6-14 — Properties of the union CD_SecondParameter

Name	Type	Card.	Description
inverseFlattening	double	0..1[1]	the inverse flattening of the ellipsoid: $f^{-1}=\frac{a}{a-b}$
semiMinorAxis	Length	0..1	The length of the semi-minor axis of the ellipsoid
isSphere	boolean	0..1	true if the ellipsoid is a sphere

Table 6-15 — Properties of the class CD_PrimeMeridian

Name	Type	Card.	Description
greenwichLongitude	Angle	1	Longitude of the prime meridian measured from the Greenwich meridian, positive eastward

[1] Exactly one member must be defined

Table 6-16 — Properties of the class CD_GeodeticDatum

Name	Type	Card.	Description
Ellipsoid	CD_Ellipsoid	1	The ellipsoid used as a model of the Earth for this datum
primeMeridian	CD_PrimeMeridian	1	The prime meridian of this datum

Table 6-17 — Properties of the class CD_ImageDatum

Name	Type	Card.	Description
pixelInCell	CD_PixelInCell	1	Specification of the way the image grid is associated with the image data attributes

Table 6-18 — Definitions of the enumeration CD_PixelInCell

Name	Description
cellCenter	The origin of the image coordinate system is the centre of a grid cell or image pixel
cellCorner	The origin of the image coordinate system is the corner of a grid cell, or half-way between the centres of adjacent image pixels

6-4.6 The Coordinate Operation package

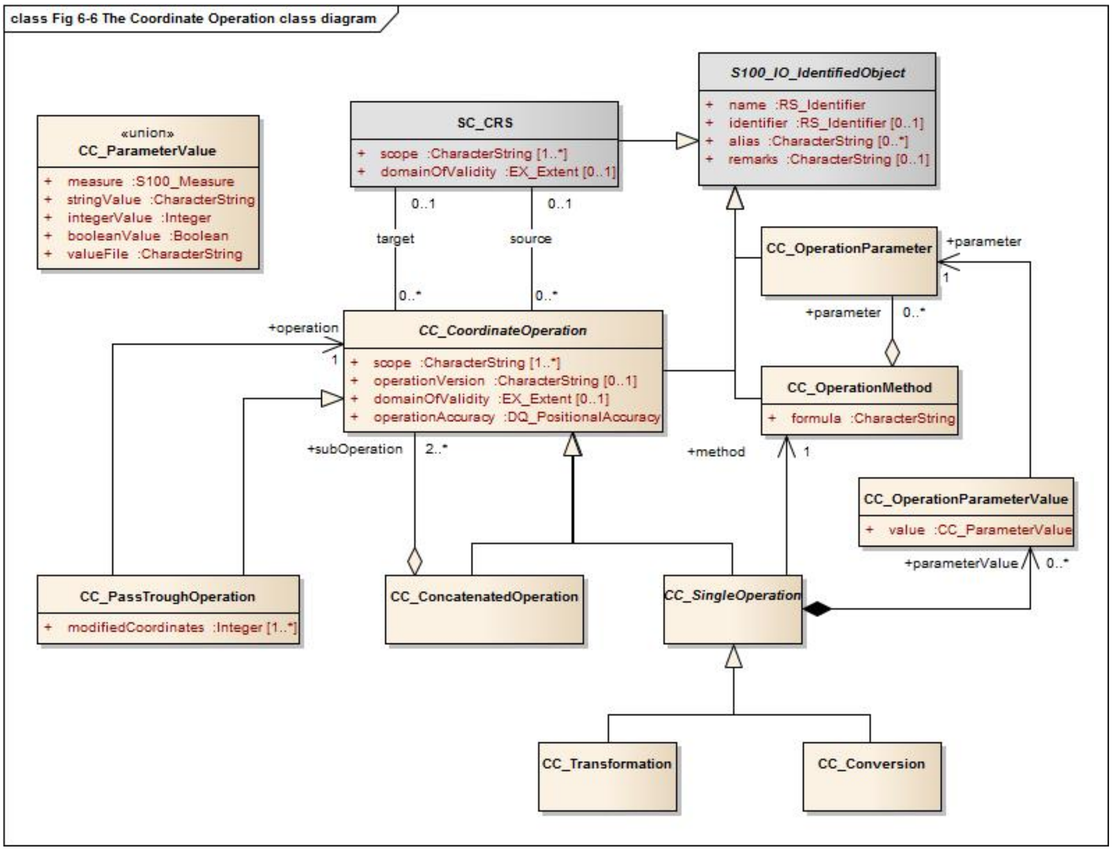


Figure 6-6 — The Coordinate Operation class diagram

Coordinate operations convert coordinates which refer to one coordinate reference system to coordinates that refer to another coordinate reference system. Therefore each coordinate operation has a source CRS and a target CRS.

The following types of coordinate operations are defined by S-100:

1) Coordinate Transformation;

2) Coordinate Conversion;

3) Pass Through Operation;

4) Concatenated Coordinate Operation.

A coordinate transformation changes coordinates from a coordinate reference system based on one datum to a coordinate reference system based on a second datum. The parameters of these operations are usually derived empirically. The stochastic nature of the parameters may result in several different versions of the same coordinate transformation. Therefore multiple coordinate transformations may exist for a given pair of coordinate reference systems, differing in their method, parameter values and accuracy characteristics.

A coordinate conversion changes coordinates between two coordinate reference systems based on the same datum. This type of coordinate operation includes map projections.
A pass through operation specifies what subset of a coordinate tuple is subject to a requested coordinate operation. It takes the form of referencing another coordinate operation and specifying a sequence of numbers defining the positions in the coordinate tuple of the coordinates affected by that coordinate operation.

EXAMPLE For a coordinate operation on the height coordinate of a tuple defined by a compound reference system the pass through operation filters the height coordinate prior to passing it to the relevant coordinate operation.

A concatenated coordinate operation is a non-repeating sequence of coordinate operations. This sequence of coordinate operations is constrained by the requirement that the target coordinate reference system of each step shall be the same as the source coordinate reference system of the next step. The source coordinate reference system of the first step and the target coordinate reference system of the last step are the source and target coordinate reference systems specified for the concatenated coordinate operation. Concatenated coordinate operation may contain coordinate conversions and coordinate transformations. If the datums of the source and target coordinate reference system are different the entire operation is a coordinate transformation.

An example of concatenation is the “Position vector 7-parameter transformation” (EPSG 9606), which is internally a concatenation of:

1) A “Geographic/Geocentric conversion” (EPSG9602);

2) A Helmert transformation on the geocentric coordinates; and

3) The inverse case of the “Geographical/Geocentrical conversion”.

Although the first and the last step are conversions that are not changing the datum, the second step does, and therefore the entire operation is a transformation.

Coordinate transformation and conversions are single coordinate operations that use similar mathematical concepts. Those concepts (algorithms or procedures) are defined by an operation method. Each operation method is fully defined by a mathematical formula and a set of parameters, although this set may be empty.

The mathematical formulas for an operation are specified in text form or by referencing a source document.

Each instance of a single coordinate operation defines a value for each parameter of the corresponding operation method. Parameters and methods are identifiable objects and may be defined by referencing.

6-4.6.1 Class details

Table 6-19— Properties of the class CC_CoordinateOperation

Name	Type	Card.	Description
scope	CharacterString	1..*	Description of usage, or limitations of usage, for which this coordinate operation is valid
operationVersion	CharacterString	0..1	Version of the coordinate transformation. Mandatory when describing a coordinate transformation, and should not be supplied for a coordinate conversion
domainOfValidity	EX_Extent	0..1	Area or region in which this coordinate operation is valid
operationAccuracy	DQ_PositionalAccuracy	0..1	Estimate of the impact of this coordinate operation on point accuracy

Table 6-20 — Properties of the class CC_SingleOperation

Name	Type	Card.	Description
method	CC_OperationMethod	1	The method (algorithm or procedure) used to perform the coordinate operation
parameterValue	CC_OperationParameterValue	0..*	A value for each parameter of the associated method

Table 6-21 — Properties of the class CC_ConcatenatedOperation

Name	Type	Card.	Description
subOperation	CC_CoordinateOperation	2..*	The ordered sequence of operations that are concatenated

Table 6-22 — Properties of the class CC_PassThroughOperation

Name	Type	Card.	Description
modifiedCoordinate	integer	1..*	Ordered sequence of positive integers defining the positions in a coordinate tuple of the coordinates affected by this pass-through operation
operation	CC_CoordinateOperation	1	The coordinate operation for which this pass through operation specifies the subset of coordinates

Table 6-23 — Properties of the class CC_OperationMethod

Name	Type	Card.	Description
formula	CharacterString	1	Formula(s) or procedure used by this operation method
parameter	CC_OperationParameter	0..*	A set of parameters used by this coordinate operation method

Table 6-24 — Properties of the class CC_OperationParameterValue

Name	Type	Card.	Description
value	CC_ParameterValue	1	Value of the coordinate operation parameter value. Most parameter values are numeric, but other types of parameter values are possible
parameter	CC_OperationParameter	1	Parameter for which the value is defined

Table 6-25 — Properties of the union CC_ParameterValue

Name	Type	Card.	Description
measure	S100_Measure	0..1[2]	A numeric value of the coordinate operation parameter with its associated unit of measure
stringValue	CharacterString	0..1	A string value of the coordinate operation parameter
integerValue	integer	0..1	An integer value of the coordinate operation parameter. Usually used for a count or index
booleanValue	boolean	0..1	A Boolean value of the coordinate operation parameter
valueFile	CharacterString	0..1	Reference to a file containing one or more parameter values. This can be a filename or an URL or some other method to reference a file

2 Exactly one member must be defined

Appendix 6-A
Examples
(informative)

Four examples are shown in this Appendix to demonstrate the use of the required information to describe a coordinate reference system:

1. 2D geodetic CRS using references to an external source;
2. Projected CRS using references to an external source;
3. The same CRS defining all details in place;
4. A compound CRS combining the first example with a vertical CRS.

An XML like notation is used for the examples. UML identifiers are used as element names. Values are shown in **bold**. For a better overview, data types may be included in the element's name and shown in blue.

6-A-1 2D geodetic CRS using references to an external source

This example uses a reference to the EPSG Geodetic Parameter Data Set. Please note that the class SC_CRS is used for the referencing and all details are defined in the referenced source. An exception is the scope since this is a mandatory field in the class SC_CRS.

```
<SC_CRS:example1>
    <RS_Identifier:name>
        <code>WGS 84</code>
    </RS_Identifier:name>
    <RS_Identifier:identifier>
        <CI_Citation:authority>
            <title>EPSG Geodetic Parameter Data Set</title>
            <edition>6.5</edition>
            <CI_Date:date>
                <date>20040113</date>
                <dateType>revision</dateType>
            </CI_Date:date>
        </CI_Citation:authority>
        <code>4326</code>
    </RS_Identifier:identifier>
<scope>
        Horizontal component of the 3D geodetic CRS used by the GPS satellite system.
</scope>
</SC_CRS:example1>
```

6-A-2 Projected CRS using references to an external source

This example is similar to A.2. It defines a projected CRS by referencing the EPSG Geodetic Parameter Data Set.

```
<SC_CRS:example2>
    <RS_Identifier:name>
        <code>Amersfoort / RD new</code>
    </RS_Identifier:name>
    <RS_Identifier:identifier>
        <CI_Citation:authority>
            <title>EPSG Geodetic Parameter Data Set</title>
            <edition>6.5</edition>
            <CI_Date:date>
                <date>20040113</date>
                <dateType>revision</dateType>
```

```
        </CI_Date:date>
    </CI_Citation:authority>
    <code>28992</code>
  </RS_Identifier:identifier>
<scope>
    Large and medium scale topographic mapping and engineering survey.
</scope>
</SC_CRS:example2>
```

6-A-3 Projected CRS defining all details

This example is the full detail of A.3.

```
<SC_ProjectedCRS:example3>
  <!-- name and scope -->
  <RS_Identifier:name>
    <code>Amersfoort / RD new</code>
  </RS_Identifier:name>
<scope>
    Large and medium scale topographic mapping and engineering survey.
</scope>

  <!-- the coordinate system -->
  <CS_CartesianCS:coordinateSystem>
    <!-- axis # 1 -->
    <CS_CoordinateSystemAxis:axis>
      <RS_Identifier:name>
        <code>Easting</code>
      </RS_Identifier:name>
      <axisSymbol>X</axisSymbol>
      <axisDirection>east</axisDirection>
      <CS_UnitOfMeasure:unitOfMeasure>
        <RS_Identifier:name>
          <code>Metre</code>
        </RS_Identifier:name>
        <symbol>m</symbol>
        <type>length</type>
      </CS_UnitOfMeasure:unitOfMeasure>
    </CS_CoordinateSystemAxis:axis>
    <!-- axis # 2 -->
    <CS_CoordinateSystemAxis:axis>
      <RS_Identifier:name>
        <code>Northing</code>
      </RS_Identifier:name>
      <axisSymbol>Y</axisSymbol>
      <axisDirection>north</axisDirection>
      <CS_UnitOfMeasure:unitOfMeasure>
        <RS_Identifier:name>
          <code>Metre</code>
        </RS_Identifier:name>
        <symbol>m</symbol>
        <type>length</type>
      </CS_UnitOfMeasure:unitOfMeasure>
    </CS_CoordinateSystemAxis:axis>
  </CS_CartesianCS:coordinateSystem>
  <!-- end of the coordinate system -->

  <!-- the coordinate conversion -->
  <CC_Conversion:conversion>
    <RS_Identifier:name>
```

```
            <code>RD New</code>
        </RS_Identifier:name>
<scope>
            Large and medium scale topographic mapping and engineering survey.
</scope>
        <!-- the operation method including the list of parameters -->
        <CC_OperationMethod:method>
            <RS_Identifier:name>
                <code>Oblique Stereographic</code>
            </RS_Identifier:name>
            <formula>See EPSG guidance No. 7</formula>
            <CC_OperationParameter:parameter>
                <RS_Identifier:name>
                    <code>Latitude of natural origin</code>
                </RS_Identifier:name>
            </CC_OperationParameter:parameter>
            <CC_OperationParameter:parameter>
                <RS_Identifier:name>
                    <code>Longitude of natural origin</code>
                </RS_Identifier:name>
            </CC_OperationParameter:parameter>
            <CC_OperationParameter:parameter>
                <RS_Identifier:name>
                    <code>Scale factor at natural origin</code>
                </RS_Identifier:name>
            </CC_OperationParameter:parameter>
            <CC_OperationParameter:parameter>
                <RS_Identifier:name>
                    <code>False easting</code>
                </RS_Identifier:name>
            </CC_OperationParameter:parameter>
            <CC_OperationParameter:parameter>
                <RS_Identifier:name>
                    <code>False northing</code>
                </RS_Identifier:name>
            </CC_OperationParameter:parameter>
        </CC_OperationMethod:method>
        <!-- The parameter value # 1 -->
        <CC_OperationParameterValue:parameterValue>
            <parameter>Latitude of natural origin</parameter>
            <CC_ParameterValue:value>
                <CC_Measure:measure>
                    <value>52° 9’ 22.1780” N</value>
                    <CS_UnitOfMeasure:uom>
                        <RS_Identifier:name>
                            <code>Degree</code>
                        </RS_Identifier:name>
                        <type>angle</type>
                    </CS_UnitOfMeasure:uom>
                </CC_Measure:measure>
            </CC_ParameterValue:value>
        </CC_OperationParameterValue:parameterValue>
        <!-- The parameter value # 2 -->
        <CC_OperationParameterValue:parameterValue>
            <parameter>Longitude of natural origin</parameter>
            <CC_ParameterValue:value>
                <CC_Measure:measure>
                    <value>5° 23’ 15.5” E</value>
                    <CS_UnitOfMeasure:uom>
                        <RS_Identifier:name>
```

```
                        <code>Degree</code>
                    </RS_Identifier:name>
                    <type>angle</type>
                </CS_UnitOfMeasure:uom>
            </CC_Measure:measure>
        </CC_ParameterValue:value>
    </CC_OperationParameterValue:parameterValue>
    <!-- The parameter value # 3 -->
    <CC_OperationParameterValue:parameterValue>
        <parameter>Scale factor at natural origin</parameter>
        <CC_ParameterValue:value>
            <CC_Measure:measure>
                <value>0.9999079</value>
                <CS_UnitOfMeasure:uom>
                    <RS_Identifier:name>
                        <code>Scale</code>
                    </RS_Identifier:name>
                    <type>scale</type>
                </CS_UnitOfMeasure:uom>
            </CC_Measure:measure>
        </CC_ParameterValue:value>
    </CC_OperationParameterValue:parameterValue>
    <!-- The parameter value # 4 -->
    <CC_OperationParameterValue:parameterValue>
        <parameter>False easting</parameter>
        <CC_ParameterValue:value>
            <CC_Measure:measure>
                <value>155000</value>
                <CS_UnitOfMeasure:uom>
                    <RS_Identifier:name>
                        <code>Metre</code>
                    </RS_Identifier:name>
                    <symbol>m</symbol>
                    <type>length</type>
                </CS_UnitOfMeasure:uom>
            </CC_Measure:measure>
        </CC_ParameterValue:value>
    </CC_OperationParameterValue:parameterValue>
    <!-- The parameter value # 5 -->
    <CC_OperationParameterValue:parameterValue>
        <parameter>False northing</parameter>
        <CC_ParameterValue:value>
            <CC_Measure:measure>
                <value>463000</value>
                <CS_UnitOfMeasure:uom>
                    <RS_Identifier:name>
                        <code>Metre</code>
                    </RS_Identifier:name>
                    <symbol>m</symbol>
                    <type>length</type>
                </CS_UnitOfMeasure:uom>
            </CC_Measure:measure>
        </CC_ParameterValue:value>
    </CC_OperationParameterValue:parameterValue>
</CC_Conversion:conversion>
<!-- end of coordinate conversion -->

<!-- the base geodetic CRS -->
<SC_GeodeticCRS:baseCRS>
    <!-- the coordinate system of the base CRS-->
```

```xml
        <CS_GeodeticCS:coordinateSystem>
            <!-- axis # 1 -->
            <CS_CoordinateSystemAxis:axis>
                <RS_Identifier:name>
                    <code>Latitude</code>
                </RS_Identifier:name>
                <axisSymbol>φ</axisSymbol>
                <axisDirection>north</axisDirection>
                <CS_UnitOfMeasure:unitOfMeasure>
                    <RS_Identifier:name>
                        <code>Degree</code>
                    </RS_Identifier:name>
                    <symbol>°</symbol>
                    <type>angle</type>
                </CS_UnitOfMeasure:unitOfMeasure>
            </CS_CoordinateSystemAxis:axis>
            <!-- axis # 2 -->
            <CS_CoordinateSystemAxis:axis>
                <RS_Identifier:name>
                    <code>Longitude</code>
                </RS_Identifier:name>
                <axisSymbol>λ</axisSymbol>
                <axisDirection>east</axisDirection>
                <CS_UnitOfMeasure:unitOfMeasure>
                    <RS_Identifier:name>
                        <code>Degree</code>
                    </RS_Identifier:name>
                    <symbol>°</symbol>
                    <type>angle</type>
                </CS_UnitOfMeasure:unitOfMeasure>
            </CS_CoordinateSystemAxis:axis>
        </CS_GeodeticCS:coordinateSystem>
        <!-- end of coordinate system of the base CRS -->

        <!-- the geodetic datum -->
        <CD_GeodeticDatum:datum>
            <RS_Identifier:name>
                <code>Amersfoort</code>
            </RS_Identifier:name>
            <scope>
                Geodetic survey, cadastre, topographic mapping, engineering survey.
            </scope>
            <CD_Ellipsoid:ellipsoid>
                <RS_Identifier:name>
                    <code>Bessel 1841</code>
                </RS_Identifier:name>
                <semiMajorAxis>6377397.155 m</semiMajorAxis>
                <CD_SecondParameter:secondParameter>
                    <inversFlattening>299.1528128</inversFlattening>
                <CD_SecondParameter:secondParameter>
            </CD_Ellipsoid:ellipsoid>
            <CD_PrimeMeridian:primeMeridian>
                <RS_Identifier:name>
                    <code>Greenwich</code>
                </RS_Identifier:name>
                <greenwichLongitude>0</greenwichLongitude>
            <CD_PrimeMeridian:primeMeridian>
        </CD_GeodeticDatum:datum>
        <!-- end of the geodetic datum -->
    </SC_GeodeticCRS:baseCRS>
```

```
    <!-- end of base geodetic CRS -->
</SC_ProjectedCRS:example3>
```

6-A-4 A compound CRS combining the first example with a vertical CRS

Here a compound CRS is defined. The horizontal component will be defined by referencing the vertical component and is defined by details (again only the vertical datum is again defined by referencing).

```
<SC_CompoundCRS:example4>
    <!-- The horizontal component -->
    <SC_CRS:component>
        <RS_Identifier:name>
            <code>WGS 84</code>
        </RS_Identifier:name>
        <RS_Identifier:identifier>
            <CI_Citation:authority>
                <title>EPSG Geodetic Parameter Data Set</title>
                <edition>6.5</edition>
                <CI_Date:date>
                    <date>20040113</date>
                    <dateType>revision</dateType>
                </CI_Date:date>
            </CI_Citation:authority>
            <code>4326</code>
        </RS_Identifier:identifier>
<scope>
            Horizontal component of the 3D geodetic CRS used by the GPS satellite system.
</scope>
    </SC_CRS:component>
    <!-- The vertical component -->
    <SC_VerticalCRS:component>
        <RS_Identifier:name>
            <code>Mean low water springs</code>
        </RS_Identifier:name>
        <scope>Hydrography</scope>
<CS_VerticalCS:coordinateSystem>
            <RS_Identifier:name>
                <code>Gravity related depth</code>
            </RS_Identifier:name>
            <!-- axis # 1 -->
            <CS_CoordinateSystemAxis:axis>
                <RS_Identifier:name>
                    <code>Depth</code>
                </RS_Identifier:name>
                <axisSymbol>z</axisSymbol>
                <axisDirection>down</axisDirection>
                <CS_UnitOfMeasure:unitOfMeasure>
                    <RS_Identifier:name>
                        <code>Metre</code>
                    </RS_Identifier:name>
                    <symbol>m</symbol>
                    <type>length</type>
                </CS_UnitOfMeasure:unitOfMeasure>
            </CS_CoordinateSystemAxis:axis>
            <!-- The vertical datum (referenced to S-57 Attribute Catalogue) -->
            <CD_VerticalDatum:datum>
                <RS_Identifier:name>
                    <code>Mean low water springs</code>
                </RS_Identifier:name>
```

```
        <RS_Identifier:identifier>
            <CI_Citation:authority>
                <title>
                    IHO TRANSFER STANDARD for DIGITAL HYDROGRAPHIC DATA - Annex
                    A
                </title>
                <edition>3.1</edition>
                <CI_Date:date>
                    <date>200011</date>
                    <dateType>publication</dateType>
                </CI_Date:date>
            </CI_Citation:authority>
            <code>VERDAT 1</code>
        </RS_Identifier:identifier>
        <scope>Hydrography</scope>
    </CD_VerticalDatum:datum>
</CS_VerticalCS:coordinateSystem>
    </SCVerticalCRS:component>
</SC_CompoundCRS:example4
```

Page intentionally left blank

S-100 – Part 7

Spatial Schema

Page intentionally left blank

Contents

Page intentionally left blank

7-1 Scope

The spatial requirements of S-100 are less comprehensive than the requirements of ISO 19107 “Geographical Information - Spatial schema” which contains all the information necessary for describing and manipulating the spatial characteristics of geographical features and on which this Part is based. Hence this Part contains only the subset of ISO 19107 classes required for S-100. This version only contains geometry, if there is a future requirement for topology then this Part will be extended to meet these requirements.

This Part specifies:

1) A subset of ISO 19107 classes (clause 6) which is the minimum required to support 0, 1, 2 and 2.5 dimensional Spatial Schemas. As such it is restricted to specifying only data and does not include operations; and
2) Additional constraints (omitted optional elements or constrained cardinalities) which are imposed on these classes by this profile.
3) Additional classes for certain kinds of curvilinear geometry. These additional classes are based on specifications that are expected to be in the next edition of ISO 19107.

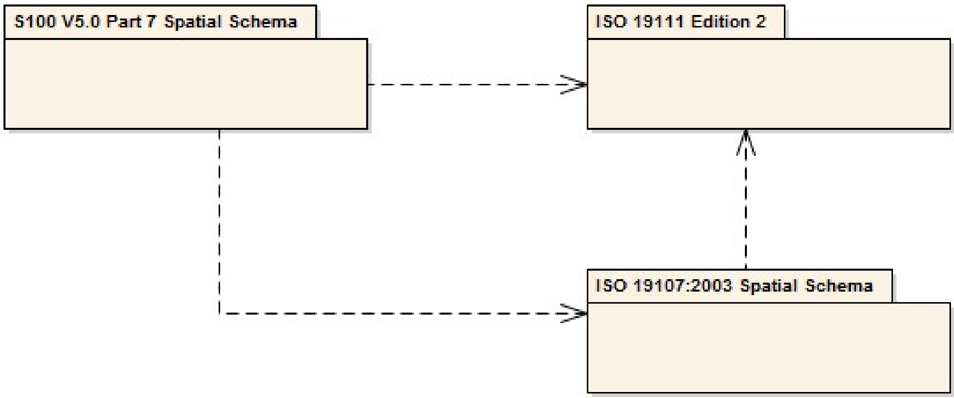


Figure 7-1 — S-100 Spatial Schema relationship with ISO 19100 Packages

7-2 Conformance

This profile consists of simple geometry based on three criteria – complexity, dimensionality and functional complexity. The first two criteria (complexity and dimensionality) determine the types defined in this profile that shall be implemented according to an Application Schema that conforms to a given conformance option.

There are:

Two levels of complexity:

1) Geometric Primitives
2) Geometric Complexes;

Four levels of dimensionality:

1) 0-dimensional objects
2) 0- and 1-dimensional objects
3) 0-, 1- and 2-dimensional objects
4) 0-, 1-, 2- and 2.5 -dimensional objects; and

One level of functional complexity:

1) Data types only (operations are not included).

This profile satisfies the conformance classes A.1.1.1, A.1.1.2, A.1.1.3, A.2.1.1 and A.2.1.2 in ISO 19107. This profile conforms to conformance class 2 of ISO 19106:2004.

7-3 References

7-3.1 Normative references

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

ISO 19107:2003, *Geographic information — Spatial schema*

ISO TS 19103:2005, *Geographic information — Conceptual schema language*

ISO 19111, *Geographic information — Spatial referencing by coordinates*

7-3.2 Non-normative references

The following references are listed only for informative purposes or to clarify parts of this document. Drafts are subject to change and are not international standards.

ISO/DIS 19107, *Geographic information – Spatial schema* (Draft – June 2018)

7-4 Geometry

7-4.1 Introduction

This profile consists of simple geometry which can be expressed in multiple configurations as described in ISO 19107:2003 clause 6.1.3.

7-4.1.1 S-100 Spatial Schema geometry classes and their ISO 19107:2003 reference

Table 7-1 Spatial types

Coordinate Geometry	Geometry Primitive	Geometry Complex	Geometry Aggregate
DirectPosition (6.4.1)	GM_Curve (6.3.16)	GM_Complex (6.6.2)	GM_Aggregate (6.5.2)
CurveInterpolation (6.4.8)	GM_CurveBoundary (6.3.5)	GM_Composite (6.6.3)	GM_MultiPoint (6.5.4)
GM_CurveSegment (6.4.9)	GM_OrientableCurve (6.3.14)	GM_CompositeCurve (6.6.5)	
GM_Position (6.4.5)	GM_OrientableSurface (6.3.15)		
GM_Polygon (6.4.36)	GM_Point (6.3.11)		
GM_SurfacePatch (6.4.34)	GM_Primitive (6.3.10)		
SurfaceInterpolation (6.4.32)	GM_Ring (6.3.6)		
S100_ArcByCenterPoint (none)	GM_Surface (6.3.17)		
S100_CircleByCenterPoint (none)	GM_SurfaceBoundary (6.3.7)		
S100_GM_SplineCurve			
S100_GM_PolynomialSpline			

7-4.1.1.1 Splines model (Informative)

The spline classes S100_GM_SplineCurve and S100_GM_PolynomialSpline in this version of S-100 bridge the curves and splines model in ISO 19107:2003 and the draft revision of ISO 19107, which is under development as this update to S-100 is being developed. Considerations in this bridging are:

- The new draft ISO model removes the concept of curve segment: "…Curve, CurveSegment, GenericCurve and CompositeCurve [are] implemented by a single class. For the same reason, there are no separate 'segments' or patches". A strict integration of this concept into S-100 would require a comprehensive overhaul of Part 7, and potentially of the S-100 data formats as well.
- The model in ISO 19107:2003 is flawed in its modelling of knots, and this has propagated into the GML Schema in ISO 19136.
- Finalization of the new edition of ISO 19107 is still some time away – the current draft is not yet an ISO International Standard and is subject to change.
- Some spline classes (or interfaces) merely add constraints and/or change a fixed attribute value compared to their generalizations, without defining any new attributes.

The cross-references for the S100 spline classes to ISO 19107:2003 and the draft ISO 19107 classes as of August 2017 are given in the table below:

Table 7-2 ISO references for S-100 spline classes

S-100 class	ISO 19107:2003 reference	Draft ISO 19107 model reference
S100_GM_SplineCurve	GM_SplineCurve (6.4.26); GM_BSplineCurve (6.4.30)	<interface>SplineCurve; <interface>BSplineCurve <datatype>BSplineData
S100_GM_PolynomialSpline	GM_PolynomialSpline (6.4.27); GM_CubicSpline (6.4.28)	<interface>PolynomialSpline; <interface>CubicSpline

All the "classes" in the draft revision of ISO 19107 are "interfaces", and the representation of the coordinates is an implementation decision. The new classes are therefore given an “S100_” prefix.

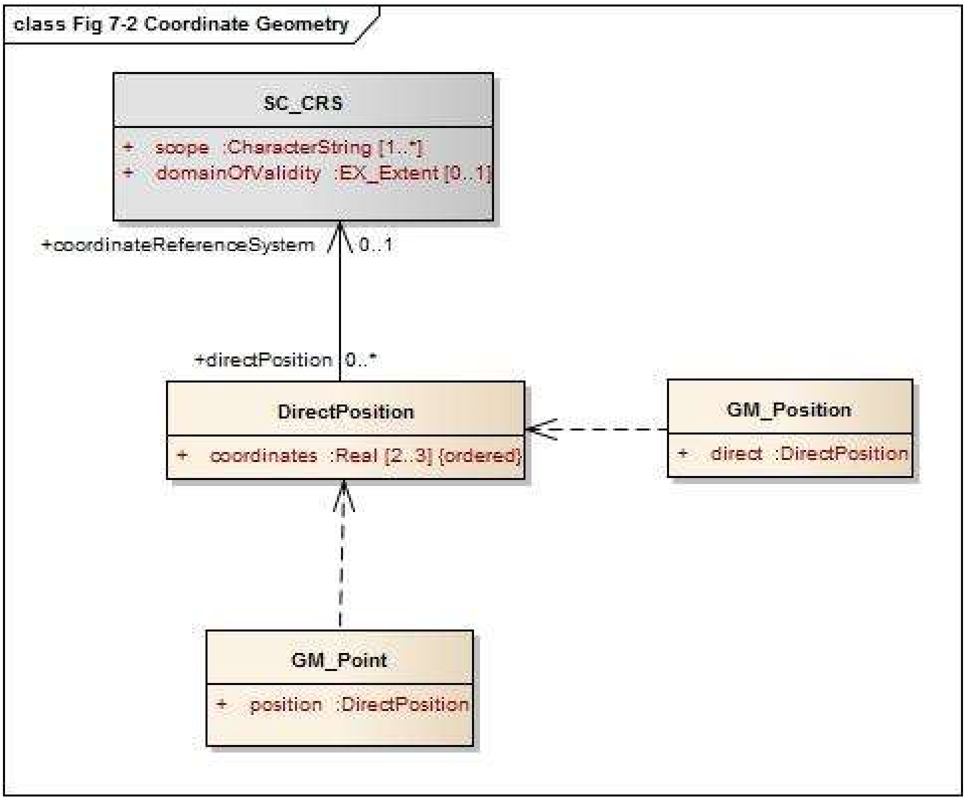


Figure 7-2 — Coordinate Geometry

7-4.1.2 DirectPosition

7-4.1.2.1 Semantics

DirectPosition holds the coordinates for a position within a particular coordinate reference system. In this profile, the associated *SC_CRS* must be linked at the *GM_Aggregate* level and not directly to a *DirectPosition*.

7-4.1.3 GM_Position

7-4.1.3.1 Semantics

The data type *GM_Position* (Figure 7-2) consists of either a *DirectPosition* or a reference to a *GM_Point* (*GM_PointRef*) from which a *DirectPosition* can be obtained.

This profile does not permit the use of the indirect position (*GM_PointRef*).

7-4.2 Simple geometry

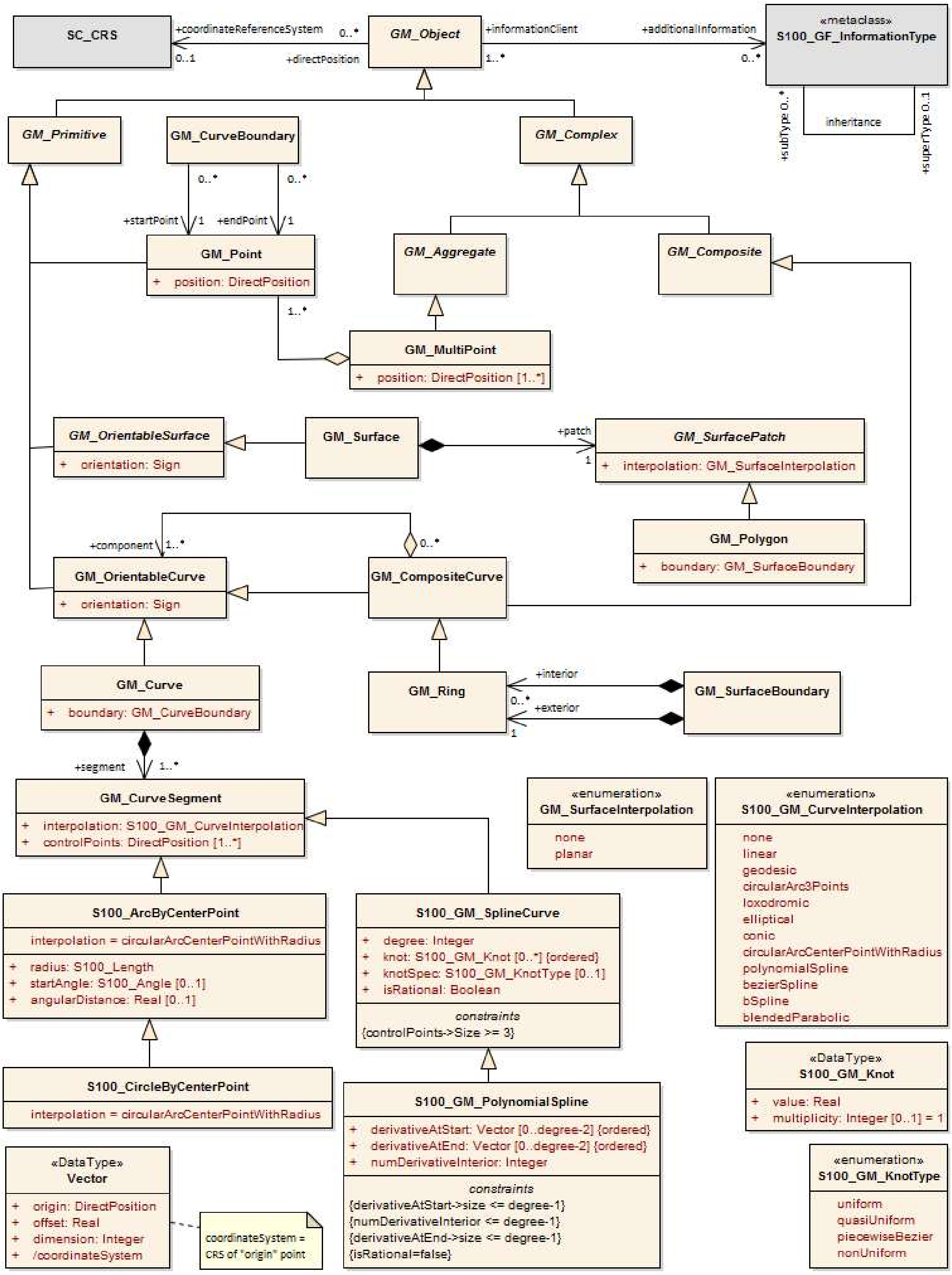


Figure 7-3 — Geometry

7-4.2.1 S100_GM_CurveInterpolation

7-4.2.1.1 Semantics

S100_GM_CurveInterpolation (Figure 7-3) is a list of codes to be used to identify the interpolation mechanisms specified by an Application Schema.

In this profile, the types of interpolation available are limited to the following:

1) None (none) – the interpolation is not specified. The assumption is that the curve conforms to the spatial object type, if constrained by that (for example, for arcs and circles) or, if not so constrained, is loxodromic.
2) Linear (linear) – the interpolation is defined by a series of DirectPositions on a straight line between each consecutive pair of controlPoints.
3) Geodesic (geodesic) – the interpolation mechanism shall return DirectPositions on a geodesic curve between each consecutive pair of controlPoints. A geodesic curve is a curve of shortest length. The geodesic shall be determined in the coordinate reference system of the *GM_Curve* in which the *GM_CurveSegment* is used.
4) Circular arc by 3 points (circularArc3Points) – the interpolation defined by a series of three DirectPositions on a circular arc passing from the start point through the middle point to the end point for each set of three consecutive controlPoints. The middle point is located halfway between the start and end point.
5) Loxodromic (loxodromic) – the interpolation method shall return DirectPositions on a loxodromic curve between each consecutive pair of controlPoints. A loxodrome is a line crossing all meridians at the same angle, that is a path of constant bearing.
6) Elliptical arc (elliptical): for each set of four consecutive controlPoints, the interpolation mechanism shall return DirectPositions on an elliptical arc passing from the first controlPoint through the middle controlPoints in order to the fourth controlPoint. Note: if the four controlPoints are co-linear, the arc becomes a straight line. If the four controlPoints are on the same circle, the arc becomes a circular one.
7) Conic arc (conic): the same as elliptical arc but using five consecutive points to determine a conic section.
8) Circular arc with centre and radius (circularArcCenterPointWithRadius) – the interpolation is defined by an arc of a circle of the specified radius centred at the position given by the single control point. The arc starts,at the start angle parameter and extends for the angle given by the angular distance parameter. This interpolation type shall be used only with S100_ArcByCenterPoint and S100_CircleByCenterPoint geometry. The precise semantics of the parameters are defined in clause 7-4.2.20 (S100_ArcByCenterPoint).
9) Polynomial (polynomialSpline) – the control points are ordered as in a line-string, but they are spanned by a polynomial function. Normally, the degree of continuity is determined by the degree of the polynomials chosen.
10) Bézier Spline (bezierSpline) – the data are ordered as in a line string, but they are spanned by a polynomial or spline function defined using the Bézier basis. Normally, the degree of continuity is determined by the degree of the polynomials chosen.
11) B-spline (bSpline) – the control points are ordered as in a line string, but they are spanned by a polynomial or rational (quotient of polynomials) spline function defined using the B-spline basis functions (which are piecewise polynomials). The use of a rational function is determined by the Boolean flag "isRational". If isRational is TRUE then all the DirectPositions associated with the control points are in homogeneous form. Normally, the degree of continuity is determined by the degree of the polynomials chosen.
12) Blended parabolic (blendedParabolic) – the control points are ordered as in a line-string, but are spanned by a function that blends segments of parabolic curves defined by triplet sequences of successive data points. Each triplet includes the final two points of its predecessor. Further details of the semantics are provided in clause 7-4.2.2.2.

7-4.2.2 GM_CurveSegment

7-4.2.2.1 Semantics

A *GM_CurveSegment* (Figure 7-3) defines the position, shape and orientation of a single *GM_Curve*. A *GM_CurveSegment* consists either of positions which are joined by straight lines, or positions which fall on a line defined by a particular type of interpolation as described in 7-5.2.1.

7-4.2.2.2 Semantics of specific interpolations

The curve interpolation type *blendedParabolic* is intended for representing smooth curves (or segments) using a reasonably low number of control points. This interpolation type means that the curve segment encoded in the control point array is composed of sequentially blended parabolic curves. The parabolic curves to be blended are determined by successive triplets of control points. Each triplet shares the two last points of its predecessor. The figure below illustrates the concept.

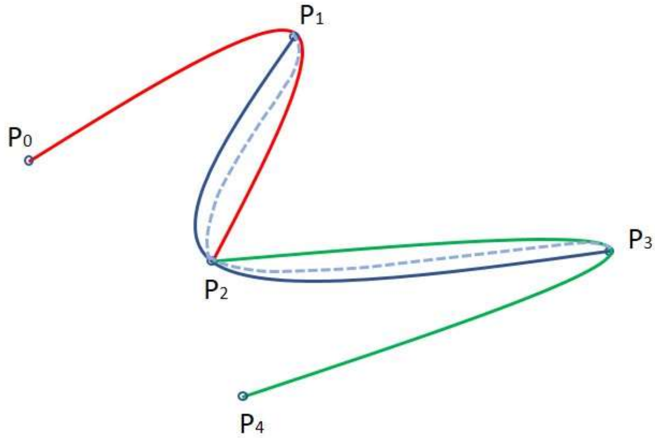


Figure 7-4 Illustration of blended parabolic interpolation

The sequence of 5 points P_0 – P_4 determines 3 parabolic segments: S1(P_0-P_1-P_2), S2(P_1-P_2-P_3) and S3(P_2-P_3-P_4). The curve between P_1 and P_2 is determined by blending the P_1-P_2 segments of S1 and S2 while the curve between P_2 and P_3 is determined by blending the P_2-P_3 segments of S2 and S3. The resultant curve between P_1 – P_3 is shown by the dashed line.

The resultant curve between two control points is computed as a combination of the two parabolas which share the two control points, for example it may be computed using the convex combination $S(P_i, P_{i+1}) = (1-t) * S_j + t * S_{j+1}$ where t varies from 0 to 1 as the path progresses from P_i to P_{i+1}.

In practice it is not necessary to compute the equations of the parabolas to be blended, as the interpolated points can be computed using the coordinates of the control points. For example, the convex combination given earlier results in the formula below for the curve from P_k to P_{k+1}, which is applied to the X and Y dimensions separately:

$$\begin{aligned} P(t) = \; & P_k + \tfrac{1}{2} t (P_{k+1} - P_{k-1}) \\ & - \tfrac{1}{2} t^2 (P_{k+2} - 4P_{k+1} + 5P_k - 2P_{k-1}) \\ & + \tfrac{1}{2} t^3 (P_{k+2} - 3P_{k+1} + 3P_k - P_{k-1}) \end{aligned}$$

For open curves the first interpolated segment of the curve segment as a whole can be generated by adding a fictitious point preceding the first point in the control point array, with coordinate values such that the second derivative in t-space (the *acceleration* of the curve) at the first point in the control point array is zero. (This allows the use of the same blending formula as for the rest of the curve.) The final interpolated segment can be computed in a similar manner by adding a fictitious control point after the last point in the array.

For closed curves, continuity and smoothness at the first and last point in the control point array require that the first triplet of control points be the same as the last triplet (or equivalently, that the curve be specifically designated as a closed curve so that the construction procedure can 'wrap around' the beginning and end of the control points array).

Due to distortions caused by applying plane methods to curved surfaces, the *blendedParabolic* interpolation should not be used where the precise location of the resultant curve is important. (It is possible to achieve higher precision by increasing the number of control points, but that defeats the pupose of using this interpolation type.)

Curves with this interpolation type have the following characteristics:

- Smooth representations with a reasonably low number of control points. However, the smoothness properties are not as high quality as cubic splines;
- Less expensive computationally than cubic splines;
- Better local control – for example, moving a control point affects only the two segments it begins and terminates and their immediate neighbors;
- There must be at least 3 points in the control points array.

7-4.2.3 GM_SurfaceInterpolation

7-4.2.3.1 Semantics

GM_SurfaceInterpolation (Figure 7-3) is a list of codes which are used to identify the method of interpolation.

In this profile, the types of *interpolation* are constrained to the following:

1) None (none) – the interior of the surface is not specified. The assumption is that the surface follows the reference surface defined by the coordinate reference system.
2) Planar (planar) – the interpolation is a section of a planar, or flat, surface. The boundary in this case shall be contained within that plane.

7-4.2.4 GM_SurfacePatch

7-4.2.4.1 Semantics

The *GM_SurfacePatch* (Figure 7-3) is the abstract root class for all 2-dimensional geometric constructs. It uses a single interpolation to define the shape and position of the associated *GM_Surface* primitives.

7-4.2.5 GM_Polygon

7-4.2.5.1 Semantics

A *GM_Polygon* (Figure 7-3) is defined by a boundary (see clause 7-4.2.7 below) and an underlying surface to which this boundary is connected. The polygon uses planar interpolation. A *GM_Polygon* is a subtype of *GM_SurfacePatch*.

7-4.2.6 GM_Curve

7-4.2.6.1 Semantics

GM_Curve (Figure 7-3) is a descendent subtype of *GM_Primitive* through *GM_OrientablePrimitive*. It is the basis for 1-dimensional geometry. A curve is a continuous image of an open interval and so could be written as a parameterized function such as $c(t):(a, b)\rightarrow E_n$ where "t" is a real parameter and En is Euclidean space of dimension n (usually 2 or 3, as determined by the coordinate reference system). Any other parameterization that results in the same image curve, traced in the same direction, such as any linear shifts and positive scales such as $e(t) = c(a + t(b-a)):(0,1) \rightarrow E_n$, is an equivalent representation of the same curve. For the sake of simplicity, *GM_Curve* should be parameterized by arc length, so that the parameterization operation inherited from *GM_GenericCurve* (see ISO 19107 clause 6.4.7) will be valid for parameters between 0 and the length of the curve.

Curves are continuous, connected, and have a measurable length in terms of the coordinate system. The orientation of the curve is determined by this parameterization, and is consistent with the tangent function, which approximates the derivative function of the parameterization and shall always point in the "forward" direction. The parameterization of the reversal of the curve defined by $c(t):(a, b)\rightarrow E_n$ would be defined by a function of the form $s(t) = c(a + b - t):(a, b)\rightarrow E_n$.

A curve is composed of one or more curve segments. Each curve segment within a curve may be defined using a different interpolation method. The curve segments are connected to one another, with the end point of each segment except the last being the start point of the next segment in the segment list.

Individual Product Specifications may constrain the interpolation types allowed for spatial attributes.

EXAMPLE: An isobar feature is constrained to curves consisting only of segments with interpolation type *polynomialSpline* and degree 3 (that is, cubic splines).

7-4.2.7 GM_CurveBoundary

7-4.2.7.1 Semantics

The boundary of *GM_Curve* shall be represented as *GM_CurveBoundary*.

7-4.2.8 GM_OrientableCurve

7-4.2.8.1 Semantics

A *GM_OrientableCurve* (Figure 7-3) is a *GM_Curve* with an associated orientation inherited from *GM_OrientablePrimative*.

7-4.2.9 GM_OrientableSurface

7-4.2.9.1 Semantics

A *GM_ OrientableSurface* (Figure 7-3) is a *GM_Surface* with an associated orientation inherited from its *GM_OrientablePrimative* parent.

7-4.2.10 GM_Point

7-4.2.10.1 Semantics

GM_Point (Figure 7-3) is a 0-dimensional geometric primitive (*GM_Primitive*).

GM_Point is the data type for a geometric object consisting of one and only one point.

7-4.2.11 GM_Primitive

7-4.2.11.1 Semantics

GM_Primitive (Figure 7-3) is the abstract root class for all geometric primitives defined in this profile. A *GM_Primitive* is a *GM_Object*. *GM_Primitive* consists of three sub-types. *GM_Point* which is 0 - dimensional; *GM_Curve* which is 1-dimensional and *GM_Surface* which is 2-dimensional. All geometric primitives (*GM_Primitive*) must be part of at least one *GM_Aggregate* (see ISO 19107 clause 8.10.1). There is no direct link between each *GM_Primitive* and the coordinate reference system *SC_CRS* used for defining the position of the *GM_Primitive.* All *GM_Primitive* contained within a *GM_Aggregate* use the same *SC_CRS* for defining their position.

7-4.2.12 GM_Ring

7-4.2.12.1 Semantics

A *GM_Ring* (Figure 7-3) is composed of a number of references to *GM_OrientableCurves*. The endpoint of *GM_OrientableCurve* “n” is the startPoint of *GM_OrientableCurve* “n+1” and the first startpoint is coincident with the last endpoint, meaning the *GM_Ring* is closed. A *GM_Ring* must be simple, that is it does not intersect itself.

7-4.2.13 GM_Surface

7-4.2.13.1 Semantics

GM_Surface (Figure 7-3) is a subclass of *GM_Primitive* and is the basis for 2-dimensional geometry. It is a *GM_OrientableSurface* with a positive orientation.

This profile does not use instances of *GM_Surface*. A *GM_Surface* within this profile must be subtyped as a *GM_Polygon.*

7-4.2.14 GM_SurfaceBoundary

7-4.2.14.1 Semantics

The boundary of *GM_Surfaces* shall be represented as *GM_SurfaceBoundary* (Figure 7-3).

A *GM_SurfaceBoundary* consists of references to a combination of at least one exterior *GM_Ring* and zero or more interior *GM_Ring*. The rings must be closed as described in ISO 19107 clause 6.6.11.1.

7-4.2.15 GM_Complex

7-4.2.15.1 Semantics

A *GM_Complex* (Figure 7-3) is a collection of geometrically separate, simple *GM_Primitive*. If a *GM_Primitive* (other than a *GM_Point*) is in a particular *GM_Complex*, then there exists a set of primitives of lower dimension in the same complex that form the boundary of this primitive. For example a *GM_Surface* is a 2 dimensional object, its boundary consists of *GM_Curve* which are 1 dimensional.

7-4.2.16 GM_Composite

7-4.2.16.1 Semantics

A geometric composite, *GM_Composite* (Figure 7-3), is a collection of primitives which must have geometry of the same type and which could exist as a single example of that primitive. For example, a composite curve is a collection of curves which could equally be represented by a single curve. This does not apply to *GM_Point* which can only contain one point.

7-4.2.17 GM_CompositeCurve

7-4.2.17.1 Semantics

A *GM_CompositeCurve* (Figure 7-3) has all the geometric properties of a curve. A composite curve is a sequence of *GM_OrientableCurve*, each curve (except the first) begins where the previous curve ends.

7-4.2.18 GM_Aggregate

7-4.2.18.1 Semantics

The aggregates, *GM_Aggregate* (Figure 7-3) gather geometric objects. Since they will often use orientation modification, the curve reference and surface references do not go directly to the *GM_Curve* and *GM_Surface*, but are directed to G*M_OrientableCurve* and *GM_OrientableSurface*.

Most geometric objects are contained in features, and cannot be held in collections that are strong aggregations. For this reason, the collections described in this clause are all weak aggregations, and shall use references to include geometric objects.

NOTE The subclasses of *GM_OrientablePrimitive* are handled in such a manner that the reference object can link to a specific orientation of that object.

7-4.2.19 GM_MultiPoint

7-4.2.19.1 Semantics

GM_MultiPoint is an aggregate class containing only points. The association role "element" shall be the set of *GM_Point* contained in this *GM_MultiPoint*.

7-4.2.20 S100_ArcByCenterPoint

7-4.2.20.1 Semantics

An S100_ArcByCenterPoint is an arc of the circle with centre given by the single control point and radius given by the *radius* parameter. Radius is geodesic distance from the centre. The arc starts at the bearing given by the *start angle* attribute and ends at the bearing calculated by adding the value of the *angular distance* parameter to the start angle. The direction of the arc is given by the sign of the angular distance, with positive values indicating a clockwise direction with respect to an observer located vertically above the centre point. Bearings are relative to true north except that arcs centred at either pole (where true north is undefined or ambiguous) shall use the prime meridian as the reference direction.

Start angle must be in degrees and is limited to the range [0.0, 360.0]. Angular distance must be in degrees and is limited to the range [-360.0, +360.0]. The upper bound on radius varies with location and reference geoid but shall be less than the minimum geodesic distance from the position of the centre to its antipodal point. Tools or Product Specifications may impose a lower limit on radius.

7-4.2.21 S100_CircleByCenterPoint

7-4.2.21.1 Semantics

An S100_CircleByCenterPoint is a circle with centre given by the single control point and radius given by the *radius* parameter. Start angle and angular distance may be omitted. The semantics and limits of the attributes are the same as S100_ArcByCenterPoint with start angle assumed to be 0.0° and angular distance assumed to be +360.0° if not provided. If provided, angular distance must be +360.0 or -360.0.

7-4.2.22 S100_GM_SplineCurve

7-4.2.22.1 Semantics

All splines share the property that they can be represented by parametric functions that map into the coordinate system of the geometric object that they will represent. Spline Curves come in essentially two forms: interpolant and approximant.

Interpolating splines ("interpolant") pass through each of the given control points. In general, the curves are defined by their data points with extra conditions at boundary points (the data points at either end of the segment), and the level of continuity (for example, C^0 continuity at a point means the curve is connected at the point; C^1 that the segments on either side have the same first derivative at the point). A cubic spline passes through each data point, is continuous and has a smooth tangent at each point.

The second type ("approximants") only approximate the control points. These splines use sets of real valued functions which are all defined on a single common domain (for example, the interval [0.0, 1.0]); are always non-negative in their values; and always sum as a complete set to 1.0 for their entire domain. These functions are used in vector equations so that the tracing of the curve is a weighted average. The spline curve always lies in the convex hull of the control points. Since such functions are defined in vector form, they can generally be used in any target dimension coordinate system.

Approximants have nice properties involving ease of representation, ease of calculation, smoothness, and some form of convexity. They do not usually pass through the control point, but if the control point array is dense enough, the local properties will force a good approximation of them, and will give a well-behaved curve in terms of shape and smoothness.

S100_GM_SplineCurve and its subclass(es) must have values of curveInterpolation that are appropriate to the type of curve; that is, one of polynomialSpline, bezierSpline, or bSpline as appropriate.

Due to distortions caused by applying plane methods to curved surfaces and the nature of splines and blended curve as approximations, the various spline and *blendedParabolic* interpolations should not be used where accuracy in the location of the resultant curve is important, such as defining the boundaries of restricted areas. (In principle it is possible to produce high-precision curves by increasing the number of control points, but that defeats the purpose of using these interpolation types.)

For the reasons mentioned in 7-4.1.1.1 and the omission of curveForm, this class is given an 'S100_' prefix.

7-4.2.22.2 Attributes

knot: The attribute "knot" is an array of knots, each of which define a value in the parameter space of the spline, and will be used to define the spline basis functions. The *knot* data type holds information on knot multiplicity. The parameter values in this array must be monotonic and strictly increasing; that is, each value must be greater than its predecessor.

degree: The attribute "degree" shall be the degree of the polynomials used for defining the interpolation. Rational splines will have this degree as the limiting degree for both the numerator and denominator of the rational functions being used for the interpolation.

knotSpec: The attribute "knotSpec" gives the type of knot distribution used in defining this spline. This is for information and possible implementation optimizations, and must be set according to the different construction-functions.

isRational: The attribute "isRational" indicates that the spline uses rational functions to define the curve. This is done by creating a polynomial spline on homogeneous coordinates, and projecting back to regular coordinates when all calculations are done. The attribute "isRational" must be "TRUE" if and only if the control points of the spline are in homogeneous coordinates, each point having a weight.

The ISO 19107 attribute "curveForm" is not used since it is for information only, used to capture the original intention.

7-4.2.22.3 Semantics of specific varieties

A B-spline is a piecewise parametric polynomial or rational curve described in terms of control points and basis functions. If the knotSpec is not present, then the knotType is uniform and the knots are evenly spaced, and except for the first and last have multiplicity = 1. At the ends the knots are of multiplicity = degree+1. If the knotType is uniform they need not be specified. B-splines must have curveInterpolation set to *bSpline*. The basis functions for B-splines depend on the degree and are defined in textbooks in mathematics, computer graphics, and computer-aided geometric design.

A B-spline curve is a piecewise Bézier curve if it is quasi-uniform except that the interior knots have attribute multiplicity[1] = "degree" rather than having multiplicity one. In this subtype the knot spacing shall be 1.0, starting at 0.0. A piecewise Bézier curve that has only two knots, 0.0, and 1.0, each of multiplicity (degree+1), is equivalent to a simple Bézier curve.

Bézier splines are polynomial splines that use Bézier or Bernstein polynomials for interpolation purposes. These polynomials are defined in textbooks in mathematics, computer graphics, and computer-aided geometric design. Bézier splines must have curveInterpolation set to *bezierSpline*.

7-4.2.23 S100_GM_PolynomialSpline

7-4.2.23.1 Semantics

A polynomial spline is a polynomial curve passing through the points in the control points array. Construction of such a spline depends on the constraints, which may include:

- restrictions on values or derivatives of the spline at the data points;
- restrictions on the continuity of various derivatives at chosen points;
- degree of the polynomial in use.

A polynomial spline of degree *n* shall be defined piecewise between knot parameter values, as an n-degree polynomial, with up to C^{n-1} continuity at the control points where the defining polynomial may change.

This level of continuity shall be controlled by the attribute numDerivativesInterior, which shall default to (degree-1).

Constructive parameters may include constraints for as many as "degree – 1" derivatives of the polynomials at each knot.

The major difference between the polynomial splines, the B-splines (basis splines) and Bézier splines is that polynomial splines pass through their control points, making the control point and sample point array identical.

7-4.2.23.2 Attributes

derivativeAtStart, derivativeAtEnd (vector): The attribute "derivativeAtStart" shall be the values used for the initial derivatives (up to degree – 2) used for interpolation in this curve at the start point of the spline. The attribute "derivativeAtEnd" shall be the values used for the final derivative (up to degree – 2) used for interpolation in this curve at the end point of the spline. These attributes are used to ensure continuity and smoothness with predecessor and successor curves if any; for example, if this curve segment is one of a sequence of curve segments, or if the curve is part a composite curve.

numDerivativesInterior (Integer): The attribute "numDerivativesInterior" is the number of continuous derivatives required at interior knots (that is, between the first and the last knot). The attribute "numDerivativesInterior" specifies the type of continuity that is guaranteed interior to the curve. The value of "0" means C^0 continuity (which is a mandatory minimum level of continuity), the value "1" means C^1 continuity, etc.

7-4.2.23.3 Semantics of specific varieties

Cubic splines are polynomial splines with degree = 3. The number of points in the control points array must be 3*N+1 where N is the number of cubic pieces.

[1] This is the attribute named "multiplicity" of class GM_Knot

7-4.2.24 S100_GM_Knot (DataType)

7-4.2.24.1 Semantics

The knots are values from the domain of a constructive parameter space for curves, surfaces and solids[2]. Each knot sequence is used for a dimension of the parameter space $k_i = \{u_0,u_1,u_2...\}$. Thus, in a surface using a functional interpolation such as a B-spline, there will be two knot-sequences, one for each parameter, $k_{i,j} = (u_i, v_j)$.

In the knot sequence for a B-spline, a knot can be repeated (affecting the underlying spline formulae). In other curves, knots will all be multiplicity 1. In S-100 knot sequences are represented as in the ISO 19107 (2017 draft) model; that is, distinct values accompanied by a multiplicity, expressed as $k_i = (t \in \mathrm{R}, m \in \mathrm{Z})$. The alternative storage form (simple sequence, with repetitions or each knot according to its multiplicity) is acceptable in a data format if required by the encoding standard on which the data format is based.

7-4.2.24.2 Attributes

value (Real): The attribute "value" is the value of the parameter at the knot of the spline. The values of successive knots must be monotonically increasing.

Multiplicity (Integer): The attribute "multiplicity" is the multiplicity of the knot.

7-4.2.25 S100_GM_KnotType

7-4.2.25.1 Semantics

A B-spline is uniform if and only if all knots are of multiplicity one and they differ by a positive constant from the preceding knot. A B-spline is quasi-uniform if and only if the knots are of multiplicity (degree+1) at the ends, of multiplicity one elsewhere and they differ by a positive constant from the preceding knot. This enumeration is used to describe the distribution of knots in the parameter space of various splines. Possible values are:

1) Uniform (uniform): Knots are equally spaced, all multiplicity 1.
2) Non-uniform (nonUniform): Knots have varying spacing and multiplicity.
3) Quasi-Uniform (quasiUniform): The interior knots are uniform, but the first and last have multiplicity one larger than the degree of the spline (p+1).
4) Piecewise Bézier (piecewiseBezier): The underlying spline is formally a Bézier spline, but knot multiplicity is always the degree of the spline except at the ends where the knot degree is (p+1). Such a spline is a pure Bézier spline between its distinct knots.

7-4.2.26 Vector

The datatype “Vector” must be associated with a point on the GeometricReferenceSurface (for example, the surface of the geoid) to be well defined. The attributes of the vector also specify the "start position" of the vector.

7-4.2.26.1 Attributes

origin: DirectPosition – The attribute “origin” is the location of the point on the GeometricReferenceSurface for which the vector is a tangent. The direct position is associated with a coordinate system; this determines the coordinate system for the vector. The direct position's spatial dimension determines the dimension of the vector.

offset: Real [1..*] – The attribute “offset” uses the coordinate system of the direct position and represents the local tangent vector in terms of the differentials of the local coordinates. The offset values are the magnitude of the vector along each coordinate axis.

dimension: Integer – The attribute “dimension” is the dimension of the origin and therefore the dimension of the local tangent space of the vector.

coordinateSystem: the attribute “coordinateSystem” is the same as the coordinate system of the origin.

---

[2] Solids are not implemented in S-100.

For curve spatial types the origin will be the point at which the vector is defined; the offset will be the latitude and longitude differentials, which together indicate the magnitude and direction of the vector; the dimension will be 2 for curves with control points encoded as latitude/longitude; and the coordinateSystem being the same as that of the origin is not encoded.

7-4.3 Geometry configurations

Figure 7-3 depicts a one size fits all geometry model which can be further constrained in both dimensionality and complexity. This is broken down into 5 basic levels.

7-4.3.1 Level 1 – 0-, 1-Dimension (no constraints)

A set of isolated point and curve primitives. Curves do not reference points (no boundary), points and curves may be coincident. Areas are represented by a closed loop of curves.

7-4.3.2 Level 2a – 0-, 1-Dimension

A set of point and curve primitives with the following constraints:

1) Each curve must reference a start and end point (they may be the same).
2) Curves must not self-intersect as shown in Figure 7-5.
3) Areas are represented by a closed loop of curves beginning and ending at a common point.
4) In the case of areas with holes, all internal boundaries must be completely contained within the external boundary and the internal boundaries must not intersect each other or the external boundary. Internal boundaries may touch other internal boundaries or the external boundary tangentially (that is at one point) as shown in Figure 7-6.
5) The outer boundary of a surface must be in a clockwise direction (surface to the right of the curve) and the curve orientation positive. The inner boundary of a surface must be in a counter-clockwise direction (surface to the right of the curve) and the curve orientation negative as shown in Figure 7-7.

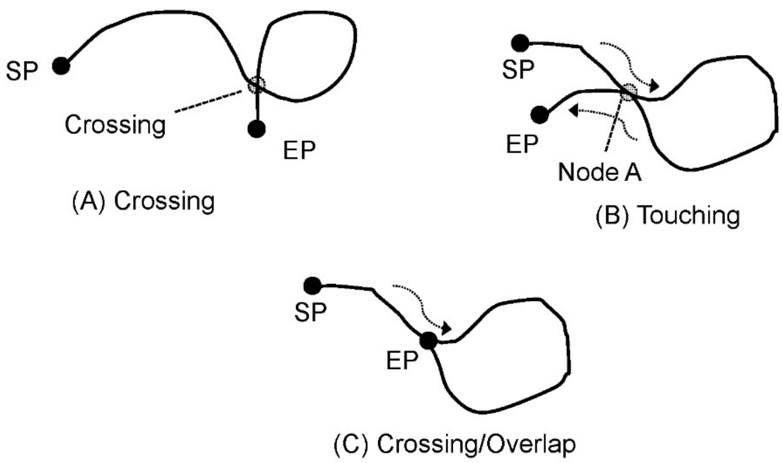


Figure 7-5 Self Intersect Example (Invalid Geometries)

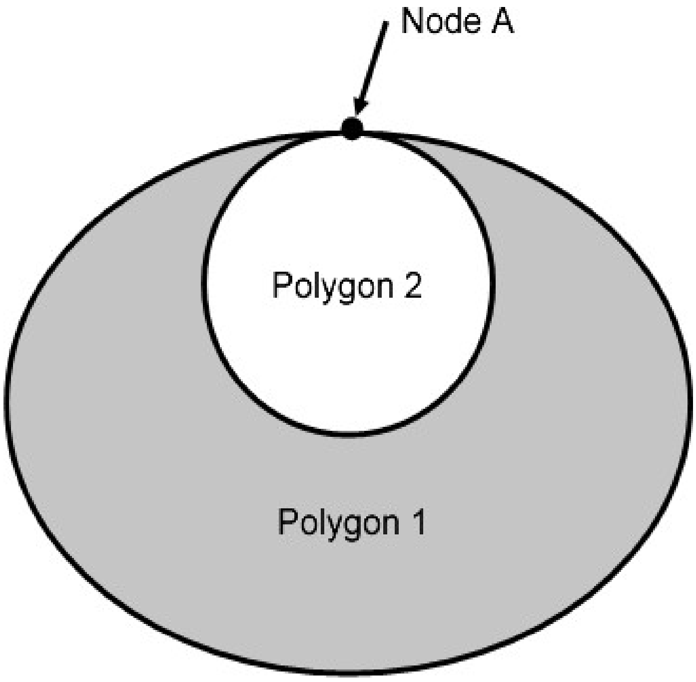


Figure 7-6 Area Holes (Valid Geometries)

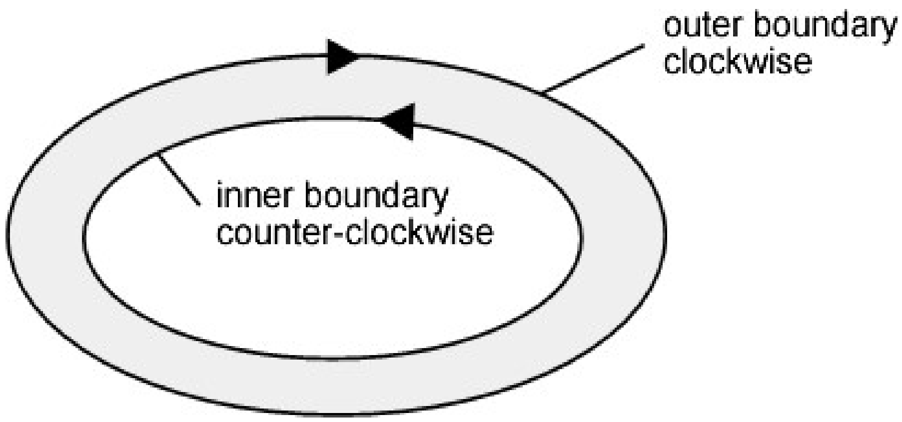


Figure 7-7 — Boundary Direction

7-4.3.3 Level 2b – 0-, 1-Dimension

A set of point and curve primitives. The constraints for Level 2a apply plus the following:

1) Each set of primitives must form a geometric complex;
2) Curves must not intersect without referencing a point at the intersection;
3) Duplication of coincident geometry is prohibited.

7-4.3.4 Level 3a – 0-, 1- and 2-Dimension

A set of point, curve and surface primitives. The constraints for Level 2a applies.

7-4.3.5 Level 3b – 0-, 1- and 2-Dimension

A set of point, curve and surface primitives. The constraints for Levels 2a and 2b apply plus the following:

1) Surfaces must be mutually exclusive and provide exhaustive cover.

Page intentionally left blank

Appendix 7-A
Examples (informative)

7-A-1 Curve Example

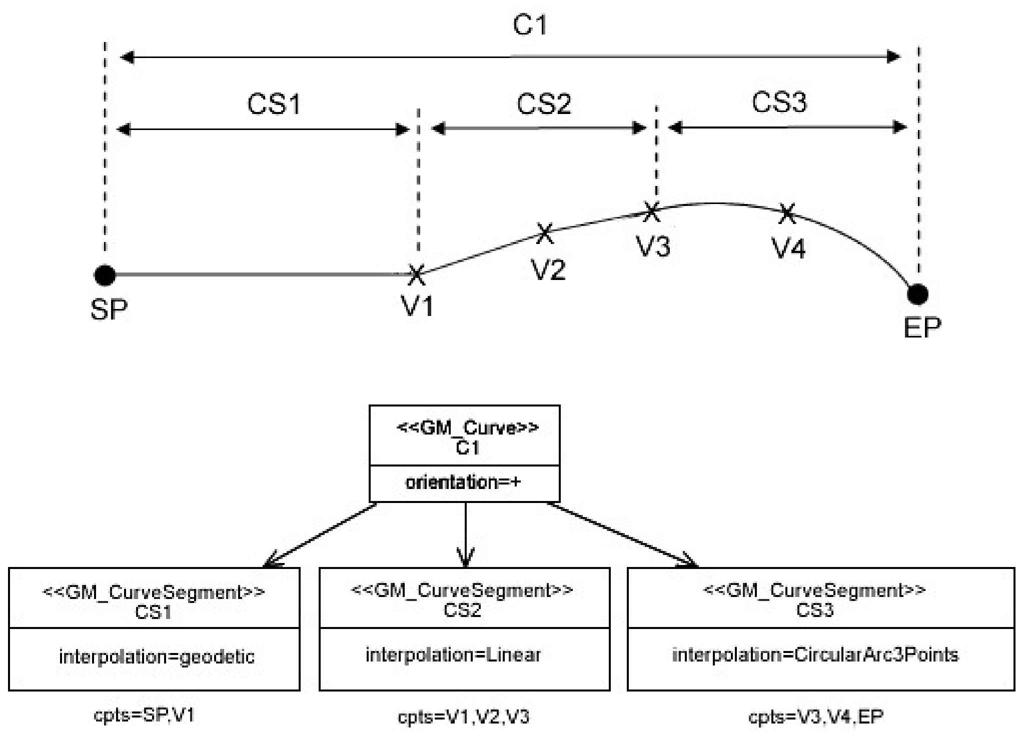


Figure 7-A-1 — Curve Example

The following describes the geometrical elements of the curve example (Figure 7-A.1).

C1 (GM_Curve) consists of CS1, CS2 and CS3 (GM_CurveSegment). CS1 uses a geodetic interpolation, CS2 linear and CS3 circularArc3Points. SP (start point) and EP (end point) (GM_Point) are the start and end points of C1 and can also be used indirectly as a 0 dimension position for a point feature. An array of control points for each segment consists of a combination of SP, EP and vertices as indicated in the above diagram. The orientation of C1 is + (forward) from SP to EP.

7-A-2 Surface Example

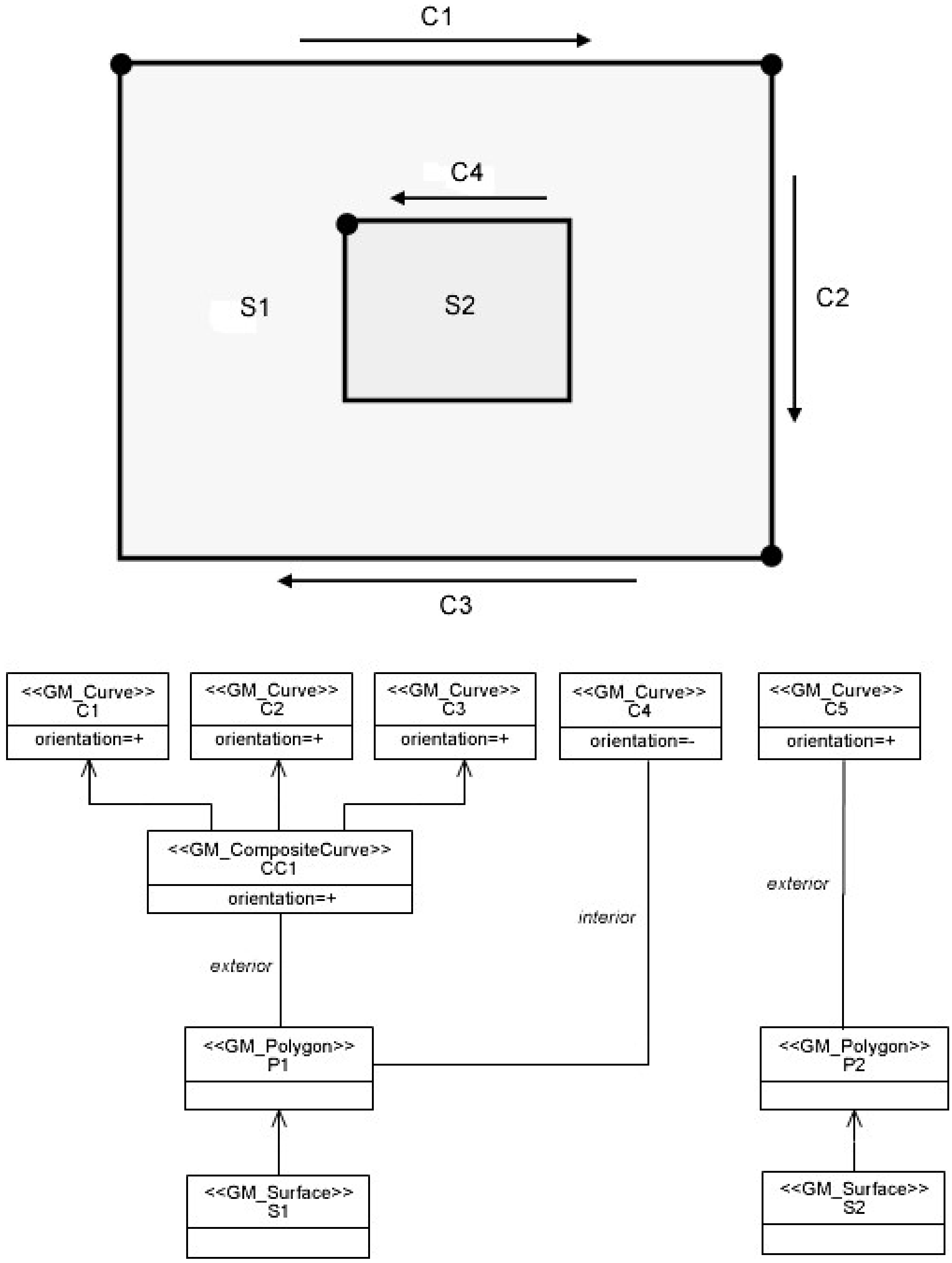


Figure 7-A-2 – Surface Example

The following describes the geometrical elements of the surface example (Figure 7-A.2).

S1 (GM_Surface) is represented by the surface patch P1 (GM_Polygon) the boundary of which consists of exterior and interior rings. The exterior ring CC1 (GM_CompositeCurve) is an aggregation of C1, C2, C3 (GM_Curve), the interior ring C4 is a simple GM_Curve.

7-A-3 2.5 Dimensional Geometry Example

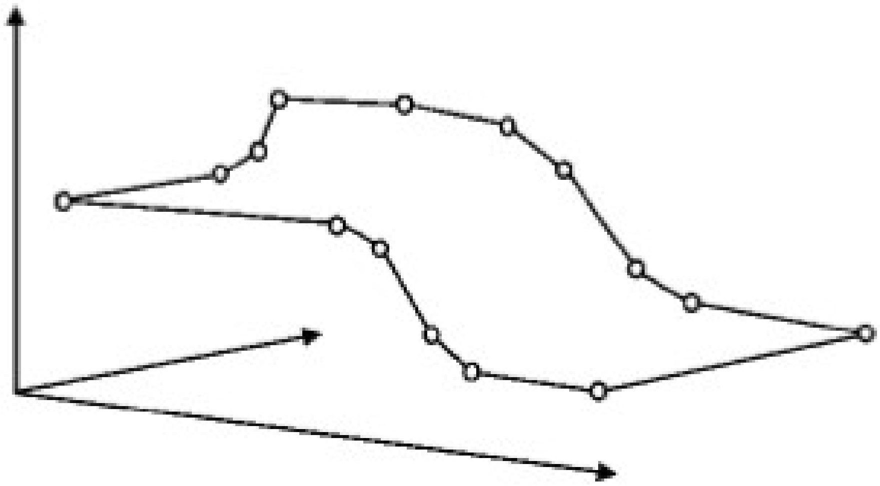


Figure 7-A-3 – 2.5D Example

In the depicted example, the curve which constitutes the exterior boundary of a GM_Polygon consists of an array of 3D control points. Note that the surface interpolation must be "none", which means that the position of interior points is not determined. The "planar" interpolation would only be acceptable if all points were lying on a plane.

Page intentionally left blank

S-100 – Part 8

Imagery and Gridded Data

Page intentionally left blank

Contents

8-1 Scope

S-100 has the capability to support imagery, gridded and several other types of coverage data as an integral component. Imagery and gridded data are common forms of geographic data and there exist many external standards designed to handle such data. An image is a particular type of gridded data structure that can be visualized. Since almost all sets of gridded data can be portrayed to form an image, the term image is very broad. S-100 must not preclude compatibility with external sources of data.

Certain kinds of hydrographic information such as soundings, current information and water level information are by their nature a set of data points distributed over an area. Other kinds of information relevant to water-borne transportation, such as meteorological information, are also data points distributed over an area. Images are also of great importance for hydrographic data. This includes images from sensors such as aerial photography or LIDAR, photographs that can be associated with vector based feature oriented data and products based on scanned paper charts, commonly known as “Raster Charts”. All of these types of data are covered by this Part of S-100.

A set of data comprised of a set of attribute values distributed over an area is called a coverage. There are many different types of coverages, for example grid structures of different types such as elevation models using a regular grid spacing; irregular grids with variable size cells; Triangulated Irregular Networks (TINs), etc.

This Part of S-100 aligns with the international standards for imagery and coverage data in order to support multiple sources of data. It uses a set of common information structures, based on the ISO TC/211 19100 suite of standards, that allows application systems to display or otherwise combine imagery, gridded and coverage data with boundary defined (vector based) data and other types of data.

This Part of S-100 defines content models and conceptual structures for grid based coverages (simple grids and selected kinds of complex grids) as well as point set coverages and TINs, for coverage data in hydrographic applications, including imagery as a type of gridded data. It describes the organization, type of grid or other coverage structure and associated metadata and spatial referencing for georeferenced data. The manner by which encoding and portrayal makes use of the content models are described in other Parts of S-100.

8-2 Conformance

8-2.1 Conformance of this Profile with other Standards

This Part defines a profile of ISO 19123:2005 concepts and types that constitutes the conceptual basis for the relevant S-100 encoding format (S-100 Part 10c — HDF5 Data Model and File Format), and for S-100-based Product Specifications describing data products based on gridded information and other coverage-type products.

This Part utilises the approach taken by ISO/TS 19129 and concepts defined in that specification to define specific conceptual structures combining ISO 19123 concepts in order to provide a simplified common basis for the S-100 encoding format for coverage-based hydrographic data, which is described in Part 10c.

8-2.2 Backward compatibility

Part 8 in this edition of S-100 is a correction, restructuring and rationalization of Part 8 in S-100 5.0.0, intended to correct internal discrepancies and update the conceptual framework to make it fully compatible with the encoding format for coverage information specified in S-100 Edition 5.0.0 Part 10c (HDF5 encoding). It simplifies Part 8 in previous editions of S-100 by removing superfluous concepts but does not entail changes to Product Specification data formats that conform to Part 10c of S-100 Edition 5.0.0.

8-2.3 Conformance to this Profile

The Abstract Test Suite presented in Appendix 8-A indicates how a coverage based product complies with the content models established in this document.

Any product addressing imagery, gridded or coverage data, claiming conformance with S-100 shall pass the requirements described in the abstract test suite, presented in Appendix 8-A.

8-3 References

8-3.1 Normative references

The following referenced documents are required for the application of this document. For dated references, only the cited edition applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IHO S-97, *IHO Guidelines for Creating S-100 Product Specifications*

ISO/TS 19103:2005, *Geographic information — Conceptual schema language*

ISO 19107:2003, *Geographic information — Spatial schema*

ISO 19111:2007, *Geographic information — Spatial referencing by coordinates*

ISO 19115-1:2018, *Geographic information – Metadata – Part 1 – Fundamentals* (as updated by Amendment 1, 2018)

ISO 19115-2:2009, *Geographic information — Metadata - Part 2 Extensions for imagery and gridded data*

ISO 19123:2005, *Geographic information — Schema for coverage geometry and functions*

ISO/TS 19129:2009, *Geographic information — Imagery, Gridded and Coverage Data Framework*

ISO 19157, Geographic information — *Data quality*

8-3.2 Informative references

The following references have been superseded by later editions or are otherwise useful though not normative.

ANSI T1.523-2001, *Telecommunications Glossary 2000, American National Standard T1.523-2001.* (Defines the term "data compaction" which is used in this Part and defined in S-100 Annex A — Terms and Definitions.)

IHO S-52, *Specifications for Chart Content and Display Aspects of ECDIS*, Edition 6.1.(1), October 2014 (with clarifications up to June 2015)

IHO S-61, *Product Specification for Raster Navigational Charts (RNC)*

IHO S-98, *Data Product Interoperability in S-100 Navigation Systems*, Edition 1.0.0, May 2022

ISO 19108, *Geographic information — Temporal schema*

ISO 19113, *Geographic information — Quality principles*

ISO 19114:2003, *Geographic information — Quality evaluation procedures.* (Superseded by ISO 19157.)

ISO 19117, *Geographic information — Portrayal*

ISO 19118, *Geographic information — Encoding*

ISO 19130:2010, *Geographic information — Sensor and data models for imagery and gridded data*

ISO/IEC 12087-5:1998, *Computer graphics and image processing -- Image Processing and Interchange (IPI) - Functional Specification - Basic Image Interchange Format (BIIF)*

ISO/IEC 15444-1:2004, *Information Technology -- JPEG 2000 image coding system*

8-4 Imagery and Gridded Data Framework

8-4.1 Framework structure

The framework for Imagery, Gridded and Coverage data used in this Part of S-100 is derived from ISO 19129 Imagery, Gridded and Coverage data Framework. Only a subset of the framework defined in the ISO standard is required in S-100[1]. The framework as described in ISO can support both georeferenced and georeferenceable data. This component of S100 is limited to georeferenced data although it can easily be extended in the future to address georeferenceable data such as sensor data.

The framework identifies how the various elements of a coverage dataset fit together. The framework provides a common structure that establishes an underlying compatibility between different sets of coverage data. The common framework established in ISO 19129 fosters a convergence at the "Content Model" level between different sets of imagery and gridded data expressed using different standards and also between the information holdings expressed using these standards. An underlying compatibility at the content model level for a broad range of imagery and gridded data allows for backward compatibility with existing standards. The content model describes information independent of the way in which it is stored, communicated or portrayed. This permits multiple encodings for the same content.

Gridded data, including imagery data, is fundamentally simple. It consists of a set of attribute values organized in a grid together with metadata to describe the meaning of the attribute values and spatial referencing information to position the data. Other coverage data is also simple. It also defines a set of points or triangles that drive a coverage function together with metadata. The metadata may contain identification information, quality information, such as the sensor from which the data was collected. The spatial referencing information contains information about how the set of attribute values is referenced to the earth. The spatial referencing information itself is expressed as metadata.

Auxiliary information, also expressed as metadata, may assist in portrayal or encoding, however the basic content may be portrayed in different ways or carried using different encoding mechanisms, so such auxiliary information is not a part of an imagery and gridded data content model. Figure 8-1 illustrates the simple structure of gridded data.

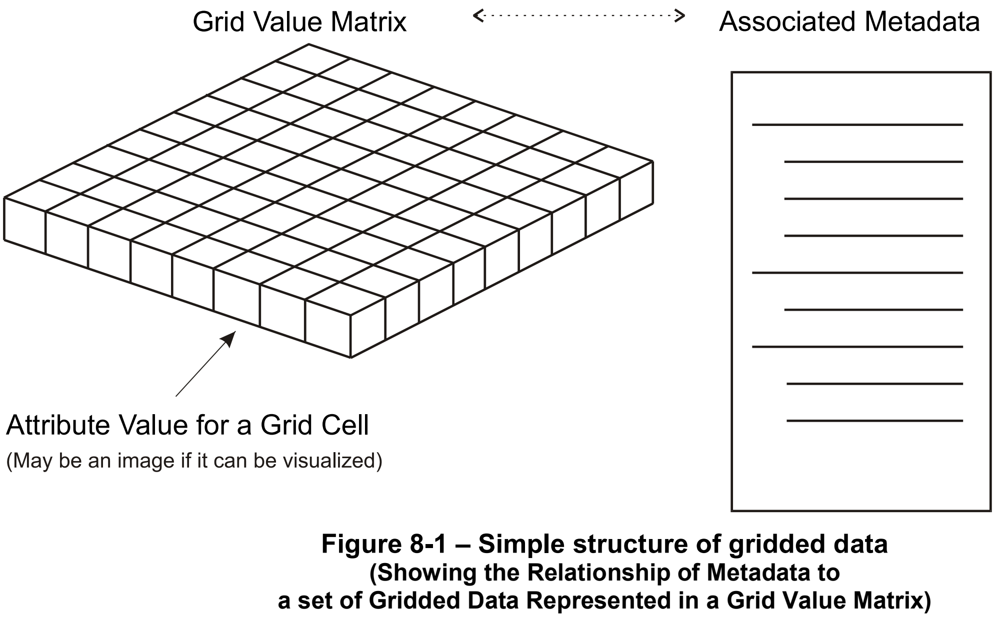


Figure 8-1 – Simple structure of gridded data
(Showing the Relationship of Metadata to
a set of Gridded Data Represented in a Grid Value Matrix)

[1] There is a commonality between the text in portions of this standard and in the ISO standard 19129 because sections of this document have been contributed to ISO as input in the development of ISO 19129 and have thus been incorporated into the ISO document.

The ISO 19129 framework standard allows Imagery, Gridded and Coverage data to be described at several levels. These are an abstract level as addressed in ISO 19123 Geographic information - Schema for coverage geometry and functions, a content model level and an encoding level. The encoding level is independent from the content level. Multiple different encodings may carry the same content.

Most of the existing standards relating to imagery and gridded data describe data content in terms of its representation in an exchange format. The format defines data fields and describes the contents and meaning of these data fields. This implicitly defines the information content that can be carried by the exchange format. Defining the content in terms of its encoding binds the content to that single encoding format and makes data conversion very difficult.

The ISO 19100 suite of standards defines geographic information content in terms of an object oriented data model expressed in the Unified Modeling Language (UML), which allows the content to be encoded using different exchange formats or stored in a database irrespective of the exchange encoding. The following figure, corresponding to ISO 19129, presents the overall relationship between the elements of the framework[2].

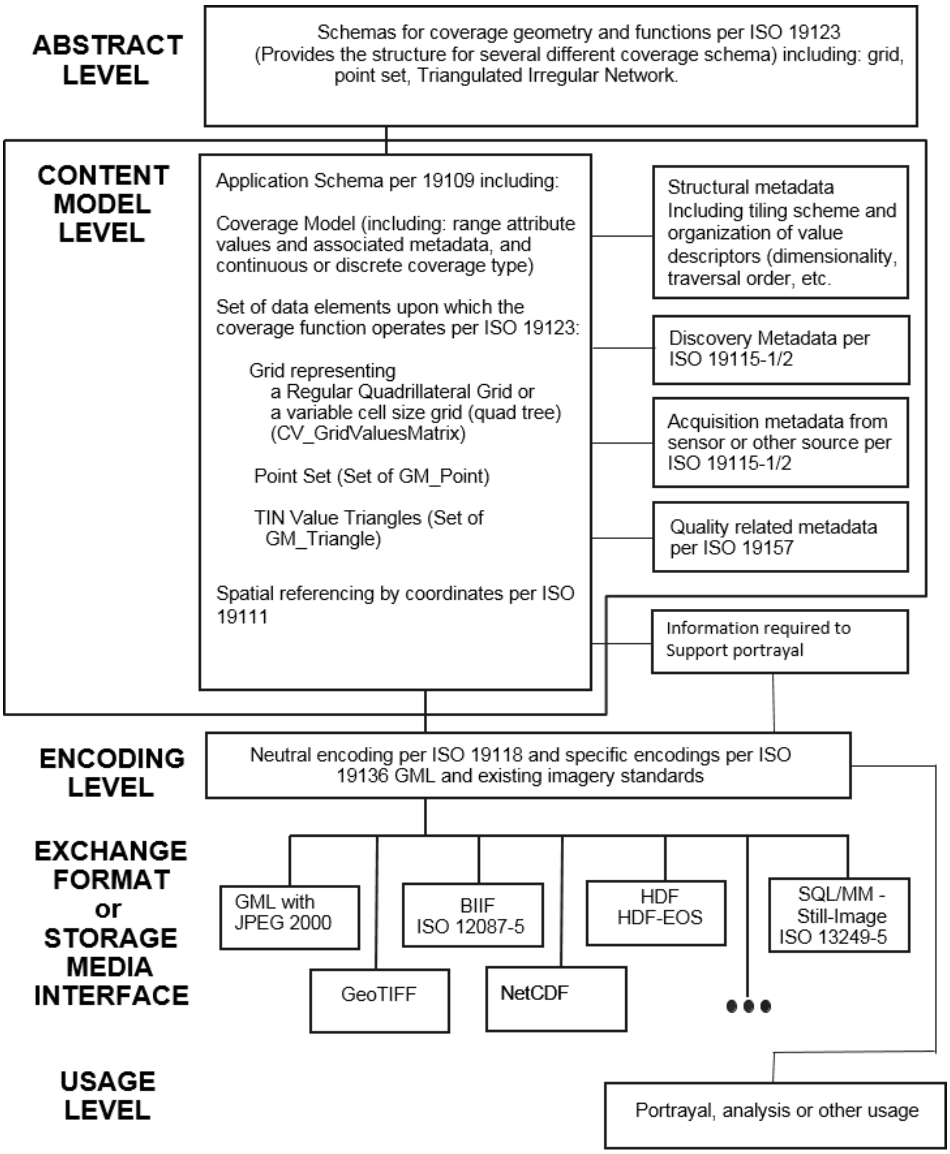


Figure 8-2 – Overall relationship between the elements of the framework

2 References to superseded ISO 191xx standards have been removed or updated.

S-100 requires certain discovery metadata elements to be provided in an Exchange Catalogue (S-100 Part 17), and permits the use of additional files conforming to the ISO 19115-3 format to encode additional metadata, including structural, acquisition, and quality metadata.

8-4.2 Abstract level

The abstract level provides a generic structure for all types of coverage geometries including gridded data geometries and point set and TIN geometries. This abstract structure is defined in ISO 19123 – Geographic information – Schema for coverage geometry and functions. S-100 takes from ISO 19123 various types of grid structures including a rectangular grid, an irregularly shaped grid, a grid with variable cell sizes and a multi-dimensional grid. A tiled grid is actually a set of grids. S-100 also includes a point coverage and a TIN coverage derived from ISO 19123.

8-4.3 Content model level

The content model level describes the information content of a set of geographic information consisting of: the Spatial Schema, feature identification and associated metadata, where other aspects such as quality, geo-referencing, etc, is represented in the metadata. The content model does not include portrayal or encoding or the organization of the data to accommodate various storage or exchange media. Exchange metadata that describes the information about a data exchange is not part of the information defined by the content model.

The content model level consists of a set of predefined content structures, which serve as the core for various Application Schemas to be developed for imagery, coverage and gridded data. This Part defines a small set of grid structures, with associated traversal orders. These provide a set of spatial organizations for gridded data. A point set structure and a TIN structure are also defined.

The feature model defined in ISO 19109 “Geographic information - Rules for Application Schema” as profiled in S-100 Part 3 applies to imagery and gridded data. The Application Schema defines one or more geographic feature types whose attributes are the elements of the respective value record. For example, a bathymetry grid may define a geographic feature type with attributes for depth and uncertainty. The geometric component for each feature type may be modelled by one of the spatial types described in this Part.

Although the conventional approach is to consider an image as a unique entity on its own, and to not consider a feature structure, it is proper to consider imagery, gridded and coverage data as feature oriented data. In the simplest form, an image or any set of gridded data can be considered as a single feature. For example, an entire satellite image could be considered as a single feature – the image. However, it is also possible to do feature extraction on an image, where sets of pixels are the geometric representation of a feature. Certain selected pixels could correspond to a bridge, and other pixels correspond to a rock. An Application Schema can contain a feature model, where the geometric component of the feature model consists of sets of geometric points corresponding to the picture elements (pixels) in a grid structure of an image. However, if a feature structure is associated with an image it is necessary to provide a method of linking feature IDs to individual pixels in the image. This can be done by carrying additional attributes in the grid value matrix, or by a pointer structure. For example, an image may be represented as a simple grid consisting of a set or rows and columns providing organization to a set of pixels. Each pixel contains attributional data such as the colour and light intensity seen at that point. Each pixel may also contain an additional attribute that indicates the feature ID associated with the pixel, so that the pixels corresponding to the image of a bridge are marked as the feature bridge, and those corresponding to a rock are marked as rock. Other more efficient structures may be defined to identify sets of pixels as corresponding to a given feature. This capability is particularly useful for adding intelligence to raster scanned image paper chart products, and for fusing S-100 vector data products with imagery and gridded data products.

Figure 8-3 below depicts the overall structure for imagery, gridded, or coverage data. The Content Model includes the spatial structure and the metadata. The encoding structure is separate but related. Systemic compression which allows for data compaction is part of the content model whereas stochastic compression which allows for data compression is not. An example of systemic compression is the removal of information that is known by the application to be not necessary. This would include areas over which there is no data (sub-tiling), and the removal of lower order bits of numeric data for lower precision numbers. A tiled grid exhibits systemic compression when tiles are only defined for areas where there is data. Systemic compression also exists in a variable size pixel structure where adjacent pixels of the same attribute value can be aggregated into a single larger pixel. Stochastic (statistical) compression removes redundant information that occurs randomly. For example, repeated bit patterns that can be compressed by an algorithm. The ZIP algorithm often used to compress files is an example of stochastic compression. Systemic compression relates to a particular type of image, whereas stochastic compression relates to a particular instance of an image. Both types of compression may be applied, but the stochastic compression is part of the encoding structure, whereas the systemic compression is part of the content model.

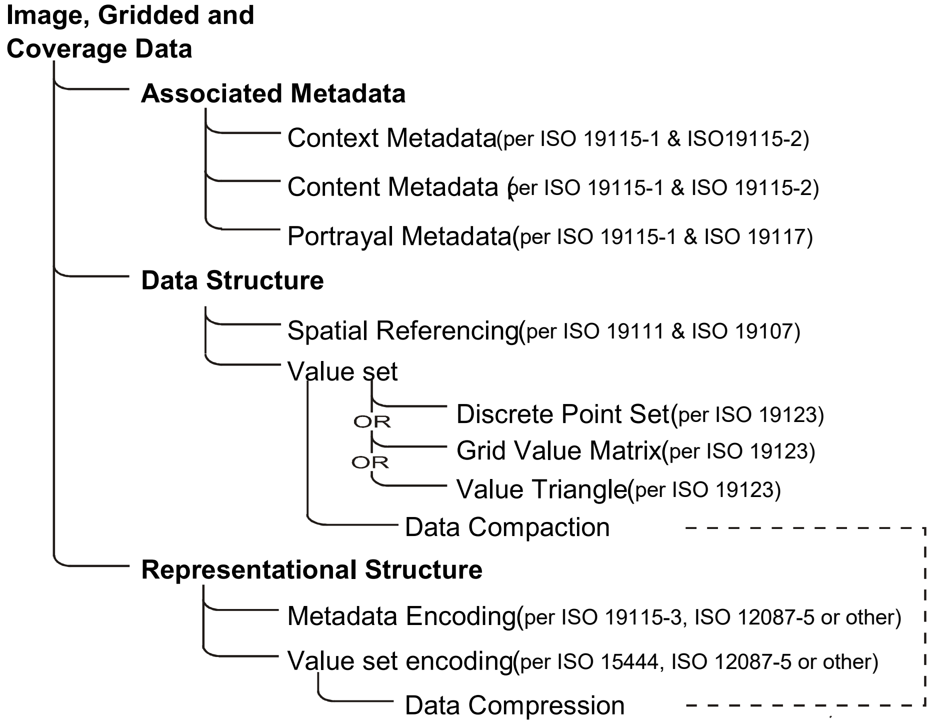


Figure 8-3 – Image and gridded data structure

Figure 8-4 below presents the elements contained in a general content model for imagery gridded and coverage data. This is a subset of Figure 8-3 above, with the representational structure not shown, since it is not part of the content model. The mechanism for systemic compression is not directly shown because it relates to the structure of the Grid Value Matrix.

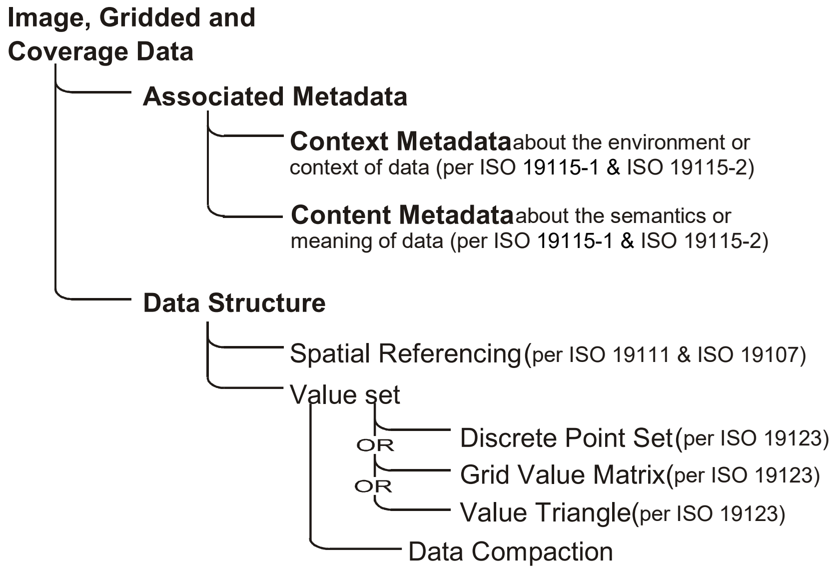


Figure 8-4 – General imagery and gridded data content description

8-4.3.1 Metadata

Typical metadata elements that are used in imagery, gridded and coverage data are presented in Table 8-1 below. The Table organizes the metadata elements according to whether the metadata relates to the description of the imagery, gridded or coverage data content, or to the environment in which it exists, or the representation of the data. Additional representational metadata may exist in an encoding format.

Table 8-1 — Metadata Elements

Type (Metadata Package)	Description	relationship
Metadata Elements (19115-1)		
Metadata information	Metadata information	Environment
Identification information	Information to uniquely identify the data. Identification information includes information about the citation for the resource, an abstract, the purpose, credit, the status and points of contact	Environment
Constraint information	Information concerning the restrictions placed on data	Environment
Data quality information (ISO 19157)	Assessment of the quality of the data	Content
Maintenance information	Information about the scope and frequency of updating data	Environment
Spatial representation information	Information concerning the mechanisms used to represent spatial information	Content
Reference system information	The description of the spatial and temporal reference system(s)	Content
Content information	Information identifying the Feature Catalogue	Content
Portrayal Catalogue information	Information identifying the Portrayal Catalogue	Representation
Distribution information	Information about the distributor of, and options for obtaining, a resource	Environment
Metadata extension information	Information about user specified extensions	Various
Application Schema information	Information about the Application Schema used to build a dataset	Content
Metadata Imagery Extensions per 19115-2		
Content Information Imagery	Additional information used to identify the content of coverage data	Content

Type (Metadata Package)	Description	relationship
Identification Information Imagery	Information to uniquely identify the data, including extensions to describe references that apply to the data and entities to identify the components used to acquire the data	Environment
Requirements Information Imagery	Provides details specific to the tasking and planning associated with the collection of imagery and gridded data	Environment
Acquisition Information Imagery	Information on the acquisition of imagery and gridded data	
Data Quality Information Imagery	Assessment of the quality of the imagery data	Content
Spatial Representation Information Imagery	Additional information the mechanisms used to represent spatial information for imagery	Content
Metadata Datatypes		
Extent information	Metadata elements that describe the spatial and temporal Extent - “geographicElement”, “temporalElement”, and “verticalElement	Content
Extent Information Imagery	Defines additional attributes used to specify the location of the minimum and maximum vertical extent values within the dataset	Content
Citation and responsible party information	A standardized method (CI_Citation) for citing a resource (dataset, feature, source, publication, etc.), as well as information about the party responsible (CI_Responsibility) for a resource	Environment

Product Specifications are not required to define metadata corresponding to the elements listed in Table 8-1. Depending on the requirements of the data product, Product Specifications may provide metadata using any combination of the following methods:

- Externally to the dataset, in the Exchange Catalogue or additional metadata files.
- Within dataset files using one or both of the following methods:
  - As attributes of the dataset, of a feature type, or individual feature instances.
  - By defining additional feature types which are also encoded in datasets as coverage features.

If the metadata are encoded as additional feature types, the Application Schema should define feature or information types defining the attributes and, for feature types, a spatial representation which must be one of the spatial types defined in this Part.

8-4.3.2 Encoding

The content model defines the structure to which an encoding rule may be applied. There are a large number of different encodings used for imagery, gridded and coverage data that provide encoding services for this class of information. Many of these encodings are well used standardized exchange formats. S-100 provides a common content model structure that can be encoded or stored using different encoding formats (for example Figure 8-2, GeoTIF).

For the coverage encoding format, S-100 uses the Hierarchical Data Format (HDF version 5), which is object oriented and suitable for all types of coverage data, including point sets and TIN triangles. The S-100-mandated profile of HDF5 is described in S-100 Part 10c.

8-5 Imagery and Gridded Data Spatial Schema

8-5.1 Coverages

8-5.1.1 General overview of coverages

A coverage associates positions within a bounded space to attribute values. A coverage is a subtype of feature; that is, it associates positions within a bounded space to the attribute values of the feature. A continuous coverage function associates a value to every position within the spatial temporal domain of the function. A discrete coverage function is only valid at specific positions within the domain. Geometric objects within the spatiotemporal domain drive the coverage function. A coverage function effectively acts as an interpolation function for the geometric objects within the spatiotemporal domain, which establishes a value within the range of the function for every position within the domain.

The geometric objects within the spatiotemporal domain are described in terms of direct positions. The geometric objects may exhaustively partition the spatiotemporal domain, and thereby form a tessellation such as a grid or a TIN. Point sets and other sets of non-continuous geometric objects do not form tessellations.

ISO 19107 defines a number of geometric objects (subtypes of the UML class GM_Object) to be used for the description of features. Some of these geometric objects can be used to define spatiotemporal domains for coverages. ISO 19123 defines additional subtypes of GM_Object that are specialised for the description of spatiotemporal domains. In addition, ISO 19108 defines TM_GeometricPrimitives that may also be used to define spatiotemporal domains of coverages.

The range of a coverage is a set of feature attribute values. The value set is represented as a collection of records with a common Schema. For example, a value set might consist of temperature and depth measured at a given time over a bounded area of ocean. A coverage function may be used to evaluate a depth and temperature anywhere within the bounded area.

The concept of coverages is described in this document to relate coverage functions to the set of geometric objects and the direct positions that drive the coverage functions. It is through the concept of coverages that one may relate the concept of features to a grid, a set of TIN triangles or a point set. This description has been adapted from ISO 19123. S-100 only addresses grids, TINs and point sets.

8-5.1.2 Discrete coverages

A discrete coverage has a spatiotemporal domain that consists of a finite collection of geometric objects and the direct positions contained in those geometric objects. A discrete coverage maps each geometric object to a single record of feature attribute values. A discrete coverage is thus a discrete or step function as opposed to a continuous coverage. For example assigning a feature code to each cell in a grid cell tessellation is a discrete coverage. Each grid cell is either associated or not associated with a particular feature. Point set coverages in hydrodynamic data, such as observations from distributed sensors, are also frequently discrete coverages in that the data values at any single point in the point set apply only in the vicinity of the point.

8-5.1.3 Continuous coverages

A continuous coverage has a spatiotemporal domain that consists of a set of direct positions in a coordinate space. A continuous coverage maps direct positions to value records. In principle, a continuous coverage could consist of no more than a spatially bounded, but transfinite set of direct positions, and a mathematical function that relates direct position to feature attribute value.

8-5.2 Point sets, grids and TINs

S-100 addresses only imagery and gridded data associated with grids, TINs and point sets. These constructs establish the basic geometry elements used in this component of S-100.

8-5.2.1 Point sets

A point set is a set of GM_Point objects in a bounded area. These point objects might each be associated with one or more features. They may also be used to form a coverage and serve to drive a coverage function. Hydrographic soundings may be considered as a point set. For each point set value it is necessary to know the position of the point as well as any associated attribute value and associated feature reference. Attributes may be assigned to an entire point set as an aggregate as well as to individual points. This is common practice for hydrographic soundings where metadata may be associated with a sounding object that consists of a point set of individual soundings. Several point sets may be aggregated into one coverage. A simple point set with associated metadata is illustrated in Figure 8-5.

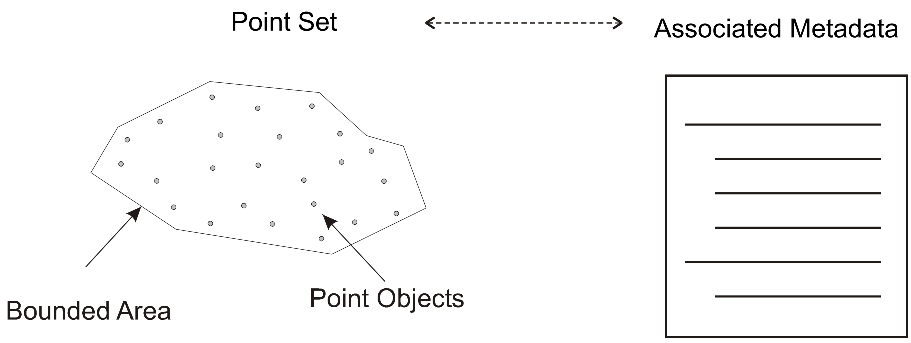


Figure 8-5 – Point Set with associated metadata

A Point Set is a set of 1, 2, 3 or n dimensional points in space. A Point Set Coverage is a coverage function associated with point value pairs in 2 dimensions. That is, a coverage function is driven by a set of points (with X, Y position) together with a record of one or more values at that position.

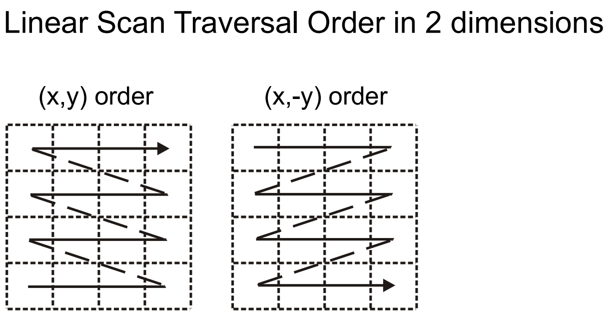
8-5.2.2 Grid types

A grid is a regular tessellation of a bounded space where two or more sets of curves in which the members of each set intersect the members of the other sets in a systematic way. The curves are called grid lines; the points at which they intersect are grid points, and the interstices between the grid lines are grid cells. A grid covers the entire bounded space. Grids form the basic geometry for a gridded data coverage. There are several different regular tessellations of a space that are all subtypes of the general concept of grid. Common to all grids is an implicit sequence or traversal order. There also exist a number of possible traversal orders for grids, some more useful than others in different situations. The location of a grid cell is defined implicitly by the regular grid organization and the traversal order. For example, in a rectangular grid each grid cell can be addressed by the row and column order of the grid. It is therefore not necessary to maintain the direct position of each grid cell. More complex grids require more complex traversal orders, however regularity still permits the position within the grid to be determined from the grid structure and the traversal order. The attribute values for a particular grid form a Grid Value Matrix where the matrix entries correspond to the grid cells.

S-100 addresses only a small subset of the possible grids and traversal orders. It makes use of the CV_DiscreteGridPointCoverage and CV_ContinuousQuadrilateralGridCoverage described in clauses 6 and 8 respectively of ISO 19123. It makes use of:

1) Rectangular grids and irregularly shaped grids;
2) Simple and tiled grids;
3) Grid with a regular cell size and variable cell sizes; and
4) Grids in 2 or 3 dimensions.

Traversal orders for grids are defined in Annex D of ISO 19123. The types of most interest to S-100 are: Linear Scan; and Morton Order. Figure 8-6 shows a linear scan traversal order and a Morton traversal order for a grid. The Morton ordering can easily accommodate irregular shaped grids, and variable cell size grids. The Morton Order corresponds to a quad tree in two dimensions but is extendable to higher dimensions.



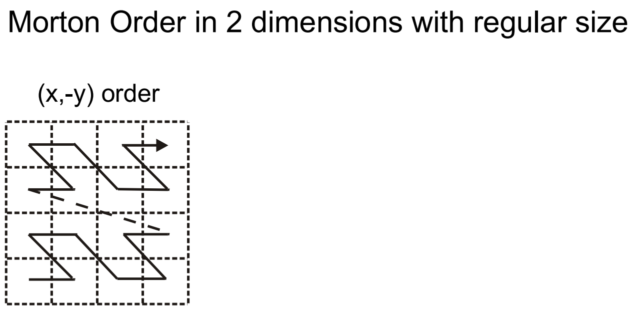


Figure 8-6 – Linear Scan Row Column (X,Y) Traversal Order and Morton (X,Y) Order

These two types of grids and traversal orders have applications for hydrographic data (for example clause 8-5.2.5 – Morton Order).

Other traversal orders defined in the ISO standard 19123 are also permitted in S-100 (see clause **Error! Reference source not found.**).

8-5.2.3 Rectangular grids and irregularly shaped grids

The most common type of grid is a rectangular grid. Most images are defined on such a grid. A rectangular grid is a subtype of quadrilateral grid as defined in ISO 19123. A quadrilateral grid is a grid in which the curves are straight lines, and there is one set of grid lines for each dimension of the grid space. In this case the grid cells are parallelograms or parallelepipeds. A parallelepiped is a three-dimensional figure like a cube, except that its faces are not squares but parallelograms.

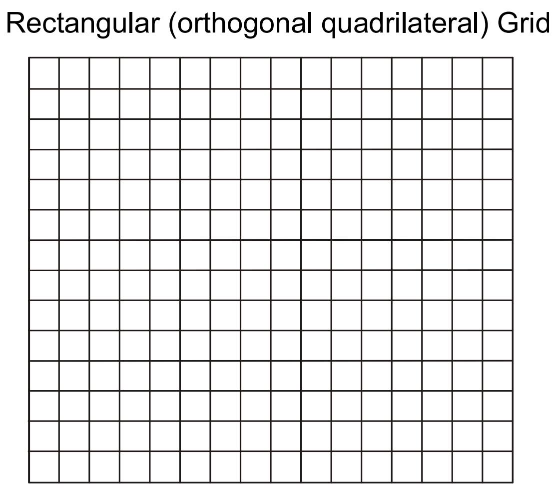


Figure 8-7 – Rectangular grid

A grid may also have a non-rectangular or quadrilateral boundary. Such grids sometimes occur when scanning paper charts that include “insets” or “outsets” that change the boundary of the grid, however the grid can have any shape, as long as it can be traversed in a sequence that gives order to the cells. Figure 8-7 shows a Rectangular Grid. Figure 8-8 shows a quadrilateral grid with an outset, as might occur in a scanning operation.

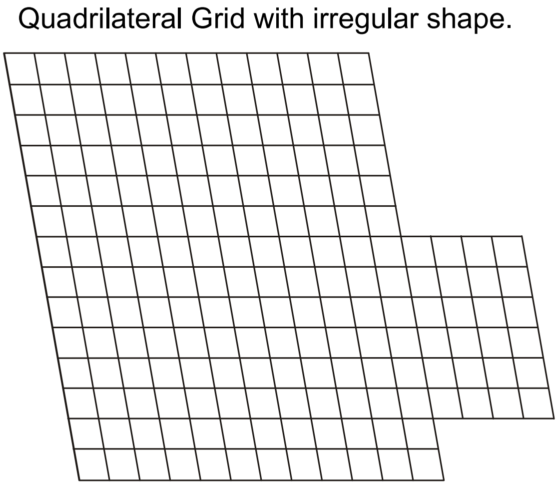


Figure 8-8 – Quadrilateral grid with outset

Very irregular shaped grids may be defined but require a more complex traversal rule than simple linear scanning.

8-5.2.4 Simple and tiled grids

A tiled grid is a combination of two or more grid tessellations for one set of data. The tiling scheme is essentially a second grid that is superimposed on the first simple grid. Each cell of the tiling scheme grid is itself a grid. A tiling scheme grid may also be used with vector data where each cell defines the boundaries of a particular vector data set. Tiling schemes are of particular value when data is sparse. For example, a raster image map of the United States might be tiled so that it is not necessary to include data over Canada or over the ocean to include Alaska and Hawaii. Figure 8-9 illustrates a tiled grid.

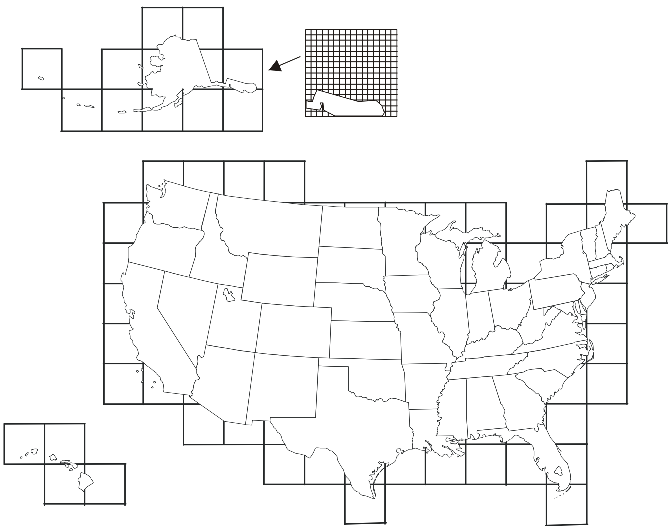


Figure 8-9 – Tiled grid

8-5.2.5 Regular and variable cell sizes

Traditional grids are fixed ‘resolution’, most commonly composed of perpendicularly crossing lines of equal spacing on each dimension, creating square or rectangular cells. Gridding is a standard way of generalizing point datasets, by imposing a resolution or grid spacing, and calculating individual grid cell values based on a single attribute of the group of points contained within each cell. As well, image data is primarily gridded, based on the resolution of the sensor or uniform arbitrary pixel spacing.

Grids may also be established where the cell size varies within the grid. A common example is the “quad tree” that is commonly used in some Geographic Information Systems. Having a variable size grid cell allows variable resolution throughout the gridded surface, which is exhibited by the unequal spacing of parallel lines that form the grid, localized to given grid cells. This requires the normalization of data on each dimension, and the binary subdivision of each dimension in order to localize any given cell. When applied to point or image data, areas of high variability can be represented by small grid cells. Areas of low variability can be represented by large grid cells. Of course if the cell size varies in a grid, it must do so in a regular way so that the grid tessellation still covers the bounded area, and the traversal method must be able to sequence the cells in an order. In addition it is necessary to include information that describes the size of each cell with the cell.

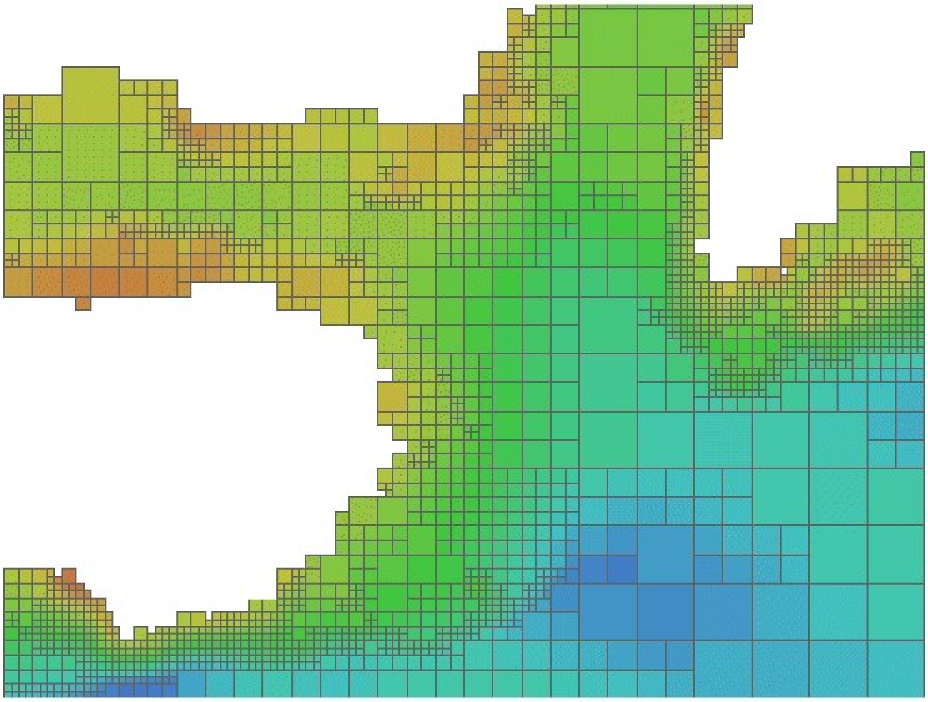


Figure 8-10 – Riemann Hyperspatial grid coverage
(showing depth from hydrographic sonar)

Data in a grid of variable cell size where adjacent like cells have been aggregated into larger cells, maintains the integrity of the original uniformly spaced data, while minimizing storage size. A grid with variable cell size supports null values, so incomplete data – that containing holes – can exist without the need to assign arbitrary values to regions of no data. This allows for a considerable amount of compaction over traditional grids because nothing is stored for cells with no data – they do not exist.

Figure 8-10 above illustrates some variable size cells. If four adjacent cells (in two dimensions) have the same attribute value in the grid value matrix, then they may aggregated into one larger cell. In two dimensions this is known as a "QuadTree". This is of particular use in applications where resolution varies, or where data values tend to cluster.

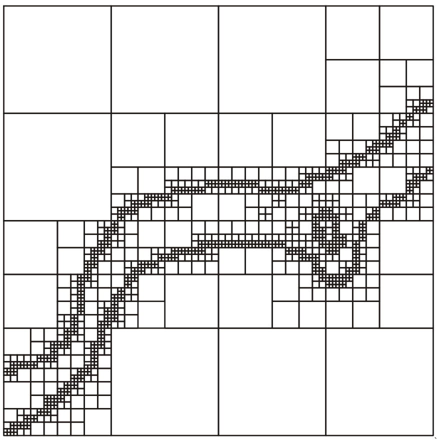


Figure 8-11 – Variable cell sizes

Variable size cells, as illustrated in Figure 8-11, are particularly useful for hydrographic data. Instead of representing bottom cover as soundings (point sets) it can be represented as a set of variable size cells. Each cell can carry several attribute values. Adjacent cells aggregate so the data volume is greatly diminished. Small cells exist where there is a rapid change in attribute value from cell to cell. Shoals, shore line and obstructions result in a number of small cells, where large relatively constant, or flat areas, such as the bottom of a channel result in a number of aggregated cells.

The Morton traversal order can handle variable size cells. The traversal progresses as shown in Figure 8-12. Morton order proceeds from left to right bottom to top cell by cell regardless of cell size. It increments in the X coordinate then the Y. This also extends to multiple dimensions where the increment is in X, then Y then Z then each additional dimension. Figure 8-13 shows Morton ordering in irregular grids and variable size grids. In this example Y, X ordering is used.

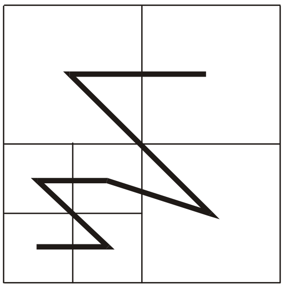


Figure 8-12 – Morton Order (X,Y)

Any space filling curve gives order to a bounded space, but the order imparted by the Morton order preserves nearness. This is a very important property. It means that two points that are close together in the grid are also close together in traversal order of the grid. This property derives from Riemann's extension of the Pythagorean Theorem into multiple dimensions into what is known as Riemann hyperspace.

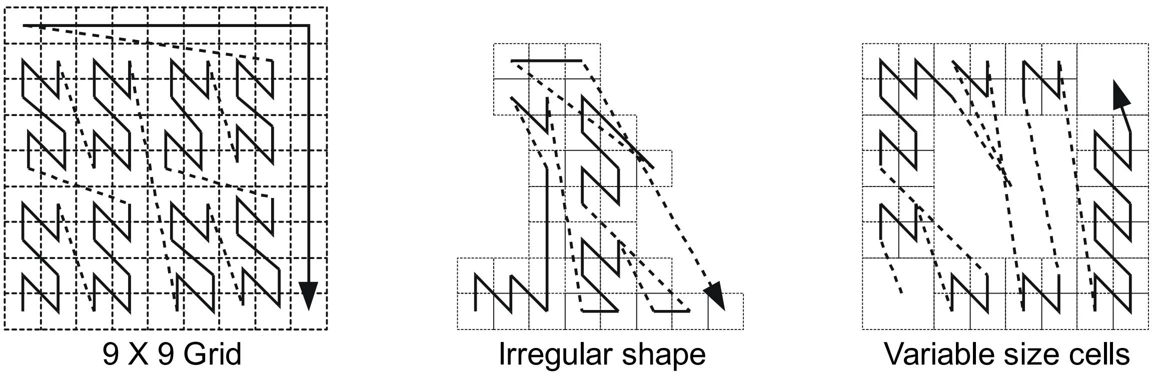


Figure 8-13 – Morton Order in irregular and variable size grids

8-5.2.6 Grids in 2 or 3 dimensions

Grids may exist in 2 or 3 dimensions. Not all traversal orders will work on higher dimensional grids, but both the linear scan traversal and Morton order traversal can be extended to 3 dimensions. Each dimension in an *n*-dimensional grid is orthogonal to all other dimensions. Thus, in a 3-dimensional grid or equal cell spacing, there are a set of perpendicularly crossing lines of equal spacing in each dimension, creating cubic cells. These can be thought of as volume elements – *voxels*.

A quadrilateral grid can easily be extended to 3 dimensions by repeating the grid for each cell "layer" in the third dimension. This is commonly done to support multiple bands of data for the same cell structure, however for true 3 dimensions where the number of cells in the third dimension is large the data volume can become enormous. Figure 8-14 shows a rectangular grid that is extended into the third dimension by repeating the grid for four different bands of data. Figure 8-15 shows a rectangular grid extended to cover a volume.

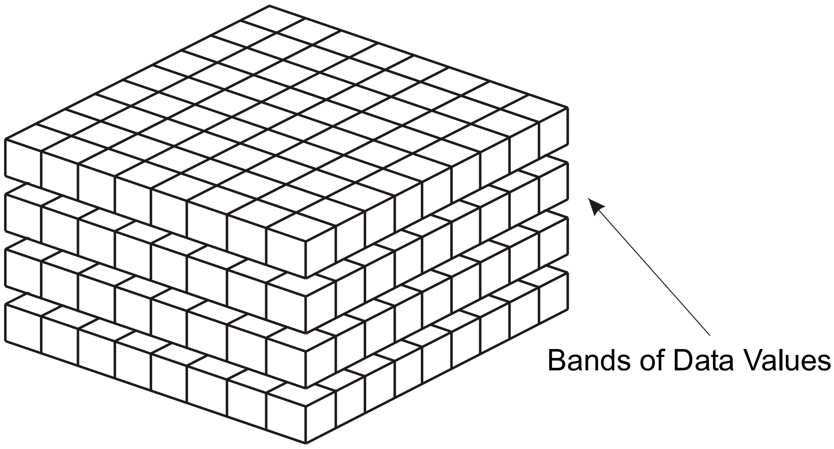


Figure 8-14 – Banding to extend attribute space in a rectangular grid

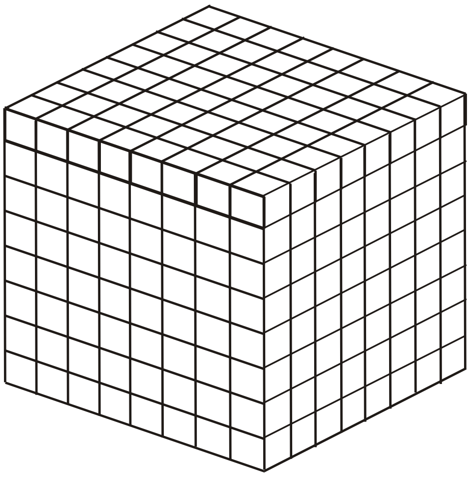


Figure 8-15 – A rectangular grid extended to cover three dimensional volume

Multidimensional Complex Grids exist in *n*-dimensions and will follow the rules of both these structures, allowing the creation for multidimensional, multi-resolution, aggregate structures. In hydrographic applications one is usually not interested in three dimensional solids but rather the three dimensional representation of the sea bottom and material, including floating material within the water volume related to the sea bottom. Such datasets are sparse, where most of the volume cells (voxels) are empty. If one allows three dimensional cells to aggregate into larger cells when they are the same (within a pre-defined tolerance), then most of the empty cells disappear into a few larger aggregations. The use of variable size cells is useful in handling three and higher dimension data. A variable size cell grid in three dimensions is illustrated in Figure 8-16.

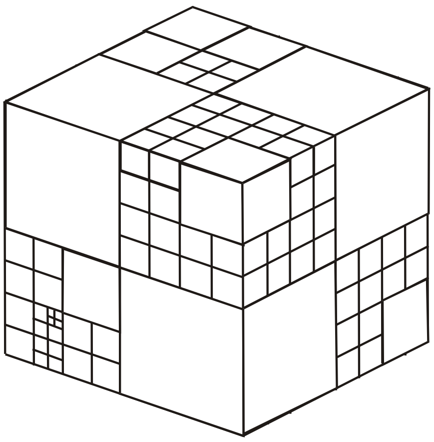


Figure 8-16 – A variable size cell grid in three dimensions

8-5.2.7 TIN

The Triangular Irregular Network is a method of describing variable density coverage data based on a set of triangles. The TIN structure is very flexible for analysis. Since each triangle is a locally flat surface it is straight forward to calculate the intersection of an arbitrary curve with a surface represented as a TIN. Attributes can be applied to each triangular face, and it is easy, but computationally intensive, to process the faces geometrically, in order to calculate contour lines. In a dynamic navigation system one could easily calculate the potential intersection of a ship's hull with the bottom surface represented as a TIN, and therefore easily determine a dynamic safe contour. The calculation of the intersection of a vector with the surface of a TIN triangle is the simple calculation of the intersection of a line and a plane. An example TIN showing variable size TIN triangles and the TIN vertex points is shown in Figure 8- 17.

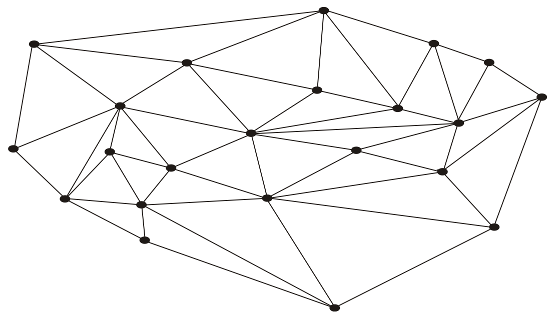


Figure 8-17 – An example coverage composed of TIN triangles

A TIN is composed of a set of triangles. The vertices at the corners of each triangle are shared with the adjacent triangle. These vertices form the control points of the coverage function. There is an inherent overhead involved in a TIN since one must store both the triangles and the vertices. Attribute values are attached to the triangles, whereas the geometry is derived from the position of the vertices. A TIN may be described either by having the triangles reference the shared vertices at their corners, or by having the vertices indicate which triangles they are attached to. Having the triangles reference the vertices is the simpler structure since each triangle has exactly 3 vertices, whereas a vertex may be shared between a variable number of triangles.

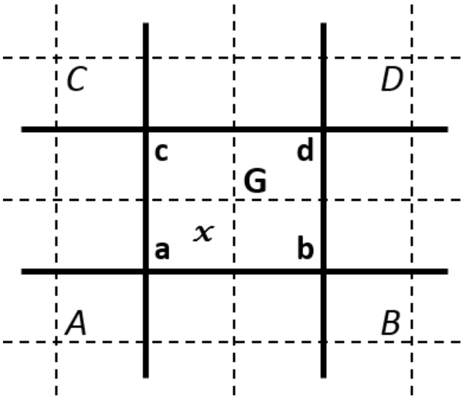
A TIN is useful in representing variable density data, since the triangles may be larger where the data is locally smooth, and more dense to represent data with more rapidly changing values. If the points of the TIN are carefully chosen to represent ridges, valleys and other significant features, then the TIN can result in a significant data compaction; however, if a TIN is automatically generated from an arbitrary set of data points the data volume can increase over the original source data, or significant information can be lost, Since a TIN coverage can be of any shape it can be fitted to cover an area of interest.

S-100 uses the CV_TINCoverage class described in ISO 19123. TIN coverages in S-100 are continuous coverages.

8-5.2.8 Grid cell structure

The feature attribute values associated with a grid point represent characteristics of the real world measured or observed within a small space surrounding a sample point represented by the grid point. The grid lines connecting these points form a set of grid cells.

EXAMPLE: In Figure 8-18 below, the grid points are (a, b, c, d), located at the intersections of the solid lines. The cells (A, B, C, D) bounded by dashed lines represent the sample spaces associated with these grid points. These sample spaces are grid cells in an offset grid (represented by the dashed lines) relative to the data grid (represented by the solid lines). Evaluation at any direct position X within the grid cell G (bounded by the solid lines) will be based on interpolation from a, b, c and d (and possibly involve additional grid points outside the cell).



Key

a, **b**, **c**, **d** grid points

G grid cell (bounded by solid lines)

A, *B*, *C*, *D* cells in sample space (bounded by dashed lines)

x direct position within the grid cell

Figure 8-18 – Grid cell structure (after ISO 19123 Figure 15)

S-100 utilizes the same view of grid cell structure as Section 8.2.2 of ISO 19123. The grid data in S-100 grid coverages are nominally situated exactly at the grid points defined by the grid coordinates. The grid points are therefore the “sample points.” Data values at a sample point represent measurements over a neighbourhood of the sample point. This neighbourhood is assumed to extend a half-cell in each dimension. The effect is that the sample space corresponding to each grid point is a cell centred at the grid point. S-100 refines the ISO 19123 view by adding optional attributes to indicate the location of the data values relative to the grid cell corners as encoded in the grid dataset. These optional attributes can be used to effectively avoid offset grids or subdivision of grid cells during evaluation and portrayal, or for more complex representations of gridded data.

Note that applying interpolation methods to a coverage means that the value of a data characteristic at a location between grid points may be different from that at any or all of the grid points which are its nearest neighbours. Such differentiation may be avoided by means of the additional S-100 attributes mentioned above.

8-5.3 Dataset structure

A dataset consists of a collection of one of more coverages together with associated metadata. Metadata may be associated with the dataset as a whole, or with individual coverages. Metadata may also be associated with particular data elements where needed. More detailed metadata at a lower level overrides general metadata for an entire coverage or collection. Metadata may also be associated with particular regions of a dataset or other grouping of dataset elements.

The description of metadata may be organized in several different ways. In this Standard the metadata is organized into modules. The S100_DatasetDiscoveryMetadata module relates to the dataset as a whole, and is described in S-100 Part 17. The S100_IF_CollectionMetadata module refers to the S100_IF_StructureMetadata module, the S100_IF_AcquisitionMetadata module and the S100_IF_QualityMetadata module as sub-components. S100_IF_CollectionMetadata is optional and may be encoded within the dataset file or externally.

Coverages or Point Set data may also be organized into tiles. Metadata may also be associated with a tile. The overall structure of a dataset is illustrated in Figure 8-19.

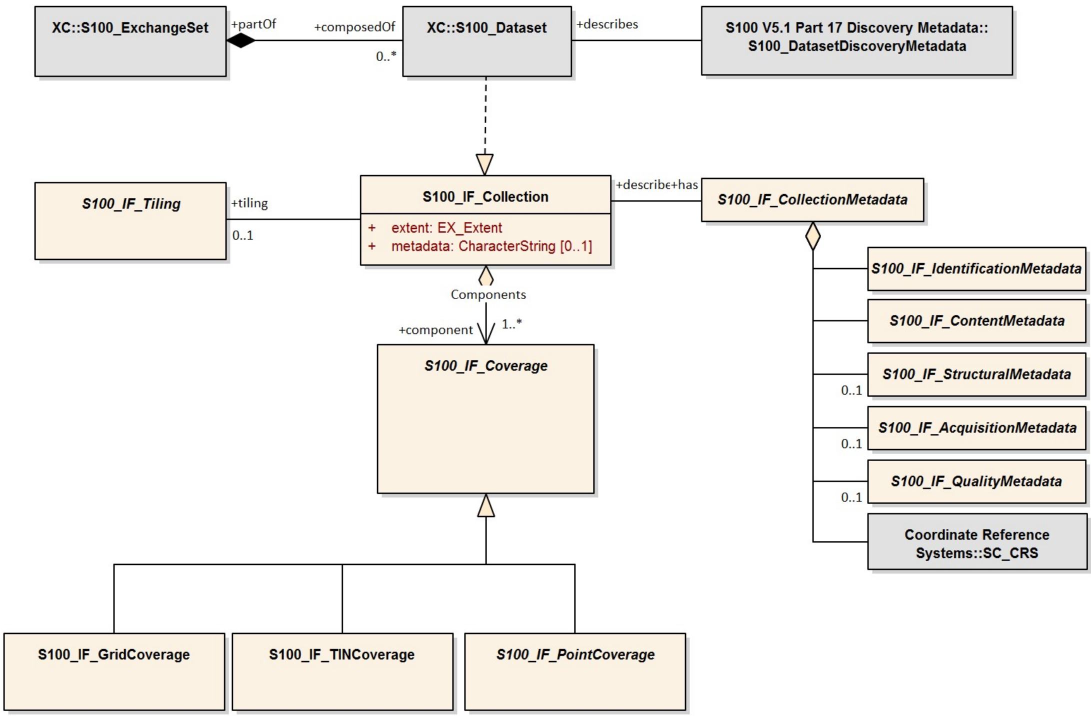


Figure 8-19 – Data Set Structure (adapted from ISO 19129:2009 Figure 7 - IGCD Structure and Metadata)

8-5.3.1 S100_Dataset

A dataset is an identifiable collection of data that can be represented in an exchange format or stored on a storage media. A dataset can represent all or a part of a logical data collection and may include one or many tiles of data. The content of a dataset is defined by the Product Specification for that particular type of data and is normally suited to the use of that data. A Product Specification for a particular data type needs to have a plan that indicates the organization of that data product. For example, a simple gridded bathymetry model based product may have only one bathymetry grid coverage, and a tiling scheme that indicates that every dataset contains one tile. More complex products may include several collocated coverages and more complex tiling schemes such as a quad tree based variable size tiling scheme, where one dataset may, at times contain more than one tile. The dataset is the logical entity that can be identified by the associated discovery metadata, not the physical entity of exchange.

8-5.3.2 S100_DatasetDiscoveryMetadata

Associated with a dataset is a set of discovery metadata that describes the dataset so that it can be accessed. It consists of the dataset discovery metadata defined in S-100 Part 17.

8-5.3.3 S100_ExchangeSet

The nominal transmittal for S-100 datasets is via Exchange Sets. An Exchange Set represents the physical entity of exchange. The transmittal is dependent upon the encoding format and the exchange media. A transmittal on a physical media such as a DVD may carry a number of datasets, whereas a transmittal over a low bandwidth telecommunications line may carry only a small part of a dataset. Any metadata carried with a transmittal is integral to the transmittal and may be changed by the exchange mechanism to other exchange metadata as required for the routing and delivery of the transmittal. A common exchange mechanism would be to carry a whole dataset on one physical media such as a CD-ROM. Transmittal metadata is not shown because any transmittal metadata, exclusive of the information in the S100_DatasetDiscoveryMetadata module, is dependent upon the mechanism used for exchange, and may differ from one exchange media or encoding format to another. An example of transmittal metadata would be counts of the number of data bytes in a unit of exchange.

8-5.3.4 S100_IF_Collection

An S100_IF_Collection represents a collection of data. A collection may include multiple different data types over a particular area, or multiple coverages of data of the same coverage type, but representing different surfaces. For example a collection may consist of a grid coverage and a point set over the same area, where the grid coverage represents a bathymetry surface and the point set a number of sounding points.

Table 8-2 — S100_IF_Collection

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_IF_Collection	Collection of data	-	-	
Attribute	extent	Spatiotemporal extent of the collection	1	EX_Extent	
Attribute	metadata	Link to metadata in external file	0..1	CharacterString	
Role	component	Coverages in the collection	1..*	S100_IF_Coverage	
Role	has	Metadata for the collection	1	S100_IF_CollectionMetadata	
Role	tiling	Tiling scheme for the collection	0..1	S100_IF_Tiling	

8-5.3.5 S100_IF_CollectionMetadata

Associated with an S100_IF_Collection is a set of collection metadata that describes the data product as represented in the collection. It consists of a number of sub-components that include S100_IF_StructuralMetadata, S100_IF_AcquisitionMetadata and S100_IF_QualityMetadata, as well as identification, coordinate reference system information, and dataset content. These metadata classes are descriptive metadata defined in ISO 19115-1, ISO 19115-2 and ISO 19157.

Table 8-3 — S100_IF_CollectionMetadata

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_IF_CollectionMetadata	Metadata for all coverages in the dataset	-	-	
Role	(component)	Identification information about the dataset	1	S100_IF_IdentificationMetadata	Product Specification, dataset issue date, etc
Role	(component)	Metadata about dataset content	1	S100_IF_ContentMetadata	Domain features and attributes
Role	(component)	Metadata about structure	0..1	S100_IF_StructuralMetadata	
Role	(component)	Metadata about acquisition	0..1	S100_IF_AcquisitionMetadata	
Role	(component)	Data quality information	0..1	S100_IF_QualityMetadata	
Role	(component)	CRS information	1	SC_CRS	Attributes to identify the CRS

8-5.3.6 S100_IF_StructuralMetadata

Associated with a data type is a set of structure metadata that describes structure of the coverage or point set.

8-5.3.7 S100_IF_AcquisitionMetadata

Associated with a data type is optionally one or many sets of acquisition metadata that describes source of the data.

8-5.3.8 S100_IF_QualityMetadata

Associated with a data type is optionally one or many sets of quality metadata that describes quality of the data.

8-5.3.9 S100_IF_Coverage

This is an abstract class used to represent all of the types of coverage or point set data that may occur in an S100_IF_Collection.

The subclasses of S100_IF_Coverage correspond to the types of coverages permitted for S-100 datasets. These classes are described in clause **Error! Reference source not found.**7.

8-5.3.10 S100_IF_Tiling

This class is an abstract class used to describe the tiling scheme used with the S100_IF_Collection. Metadata identifying a particular instance of a tile is included in the S100_IF_StructuralMetadata module. Typical tiling schemes are described in clause **Error! Reference source not found.**6.

8-5.3.11 S100_IF_IdentificationMetadata

Identification metadata for S-100 coverage datasets includes information about the Product Specification, issue data and time, geographic location identifier, etc. This is an abstract class realized in S-100 Part 10c.

8-5.3.12 S100_IF_ContentMetadata

This is an abstract class representing metadata about dataset content, in particular features and attributes as defined in the Feature Catalogue. It is realized in S-100 Part 10c.

8-5.3.13 SC_CRS

This class represents information about the coordinate reference system and datums (horizontal and vertical) used by the coverages. It is realized by attributes defined in S-100 Part 10c.

8-6 Tiling Scheme

Tiling is one method of reducing the volume of data in a dataset to manageable proportions. In a dataset there must be information both describing the tiling scheme and also about the instance of a tile or tiles carried in that particular dataset. The class S100_TilingScheme carries information about the tiling scheme as a whole. There may only be one tiling scheme defined for a particular data collection. Within a data warehouse (database) there may be several overlapping tiling schemes defined where any of the tiling schemes may be used as the basis of data extraction from the data warehouse.

A tiling scheme is itself a discrete coverage. It is normally a simple rectangular grid with tiles of equal density. Such a grid coverage may also be defined with tiles of variable density. A more complex tiling scheme may also be defined as a discrete polygon coverage. An example is a data collection consisting of elevation cut along political boundaries. These types of tiling schemes are illustrated in Figure 8-20. Other tiling schemes are also possible. In fact, any type of discrete coverage may be used to establish a tiling scheme.

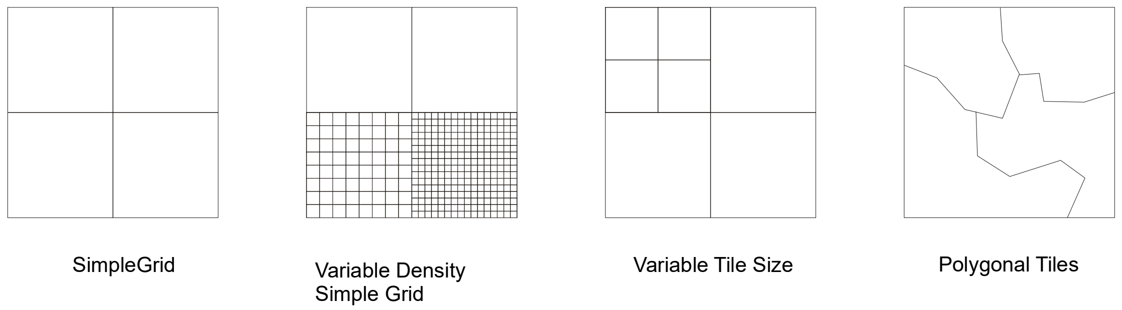


Figure 8-20 – Tiling scheme types

Any tiling scheme used must be completely described as part of the Product Specification for a particular data product. This includes the dimensions, location and data density of tiles as well as a tile identification mechanism (tileID).

8-7 Spatial Schema

Each of the S100_IF_Coverage subclasses has a specific Spatial Schema that describes the structure of that data type. The data types identified in clause 8-5.3 are:

1) S100_IF_PointCoverage:
2) S100_IF_TINCoverage; and
3) S100_IF_GridCoverage.

The conceptual models of these coverage subclasses make use of point sets and geometry/value pairs, which are described by the classes S100_IF_PointSet and CV_GeometryValuePair and its subclasses.

8-7.1 S100_IF_PointSet spatial model

An S100_Point is a single point referenced to a 3-D coordinate reference system. Its value is carried as a coordinate rather than an attribute. Such points are generated by certain types of sensors. An S100_IF_PointSet is not a coverage. A Point Set can be used to generate a Point Coverage. The class S100_IF_PointSet is illustrated in Figure 8-21.

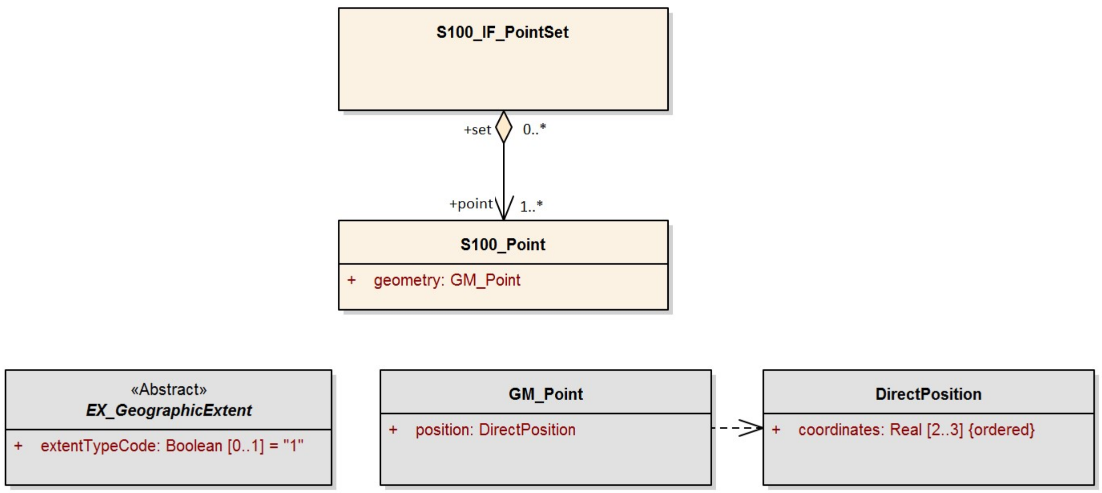


Figure 8-21 – S100_Point

The attribute *geometry* contains an instance of GM_Point.

8-7.2 CV_GeometryValuePair and subclasses

The class CV_GeometryValuePair describes an element of a set that defines the relationships of a discrete coverage. Each member of this class consists of two parts: a domain object from the domain of the coverage to which it belongs; and a record of feature attribute values from the range of the coverage to which it belongs. CV_GeometryValuePair may be generated in the execution of an evaluate operation; and need not be persistent. CV_GeometryValuePair is subclassed (see clause 8-5) to restrict the pairing of a feature attribute value record to a specific subtype of domain object.

Table 8-4 — CV_GeometryValuePair

Role Name	Name	Description	Mult	Data Type	Remarks
Class	CV_GeometryValuePair	(S-100) 2-tuple of coverage spatial element and value record for that element	-	-	In S-100 this is an abstract model of the pairing between elements of a coverage and the associated values
Attribute	geometry	Spatial primitive	1	GM_Point OR CV_GridPoint	S-100 uses only points (including TIN vertices) or grid points as spatial elements of coverages
Attribute	value	Record of feature attribute values associated with this spatial object	1	RecordType	

S-100 uses only two of the subclasses of CV_GeometryValuePair defined in ISO 19123:

- CV_PointValuePair, the subclass of CV_GeometryValuePair that has geometry = GM_Point and represents values at single points.
- CV_GridPointValuePair, the subclass of CV_GeometryValuePair that has geometry = CV_GridPoint and represents values at the grid points.

Table 8-5 — CV_PointValuePair

Role Name	Name	Description	Mult	Data Type	Remarks
Class	CV_PointValuePair	CV_PointValuePair is the subtype of CV_GeometryValuePair that has a GM_Point as the value of its geometry attribute	-	CV_GeometryValuePair	
Attribute	geometry	Spatial primitive	1	GM_Point	
Attribute	value	Value record	1	RecordType	Inherited from CV_GeometryValuePair Must conform to rangeType

Table 8-6 — CV_GridPointValuePair

Role Name	Name	Description	Mult	Data Type	Remarks
Class	CV_GridPointValuePair	CV_GridPointValuePair is the subtype of CV_GeometryValuePair that has a CV_GridPoint as the value of its geometry attribute	-	CV_GeometryValuePair	Applies only to grid coverages
Attribute	geometry	Spatial primitive	1	CV_GridPoint	
Attribute	value	Value record	1	RecordType	Inherited from CV_GeometryValuePair Must conform to rangeType

8-7.3 S100_IF_PointCoverage spatial model

An S100_IF_PointCoverage is a type of CV_DiscretePointCoverage from ISO 19123. The attribute values in the value record for each CV_GeometryValuePair represent values of the coverage, such as bathymetric soundings.

The class S100_IF_PointCoverage (Figure 8-22) represents a set of values, such as bathymetric depth values, assigned to a set of arbitrary X,Y points. Each point is identified by a horizontal coordinate geometry pair (X,Y) and assigned one or more values as attribute values. These values are organized in a record for each point.

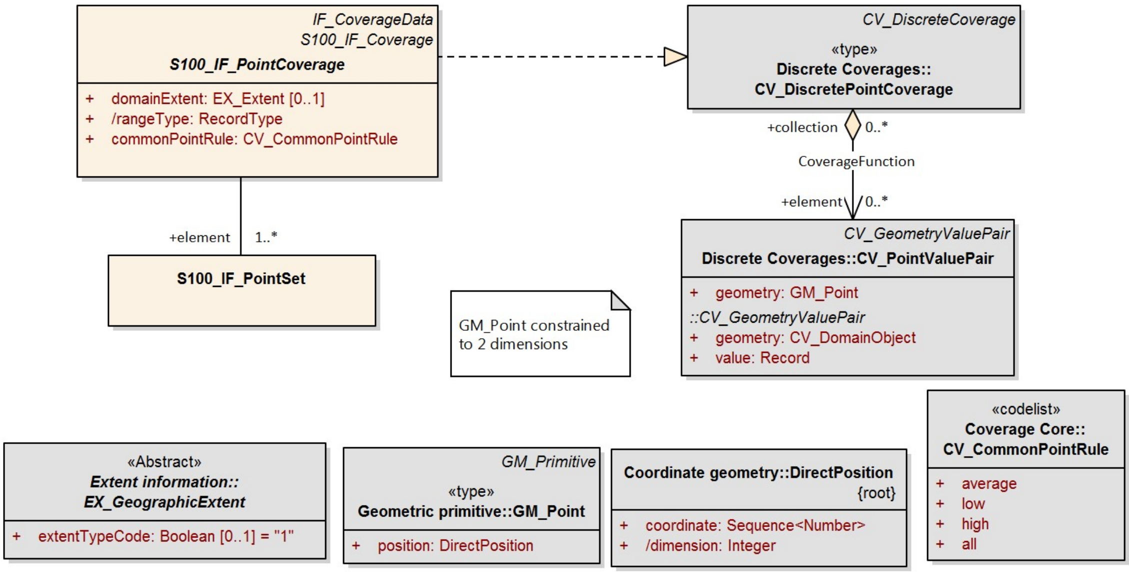


Figure 8-22 – S100_IF_PointCoverage

Table 8-7 — S100_IF_PointCoverage

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_IF_PointCoverage	Spatial type for point coverages	-	-	
Attribute	domainExtent	The spatiotemporal extent of the domain of the coverage	0..1	EX_Extent	Required only if the extent is different from the extent of the owning dataset See Note 1
Attribute	rangeType	Describes the range of the coverage A list of name:data type pairs each of which describes an attribute type included in the range of the coverage	1	RecordType	Implemented externally as Feature Catalogue (S-100 Part 5) and feature information group (S-100 Part 10c)
Attribute	commonPointRule	Procedure used for evaluating the coverage at a position that falls on the boundary or in an area of overlap between geometric objects in the domain of the coverage	1	CV_CommonPointRule	The ISO values “start” and “end” are not used See Note 2
Role	element	Spatial element	1..*	S100_IF_PointSet	

NOTE 1: *domainExtent* may be implemented as a geographic bounding box or bounding polygon with the temporal extent being implemented using separate attributes.

NOTE 2: The rule defined by attribute *commonPointRule* shall be applied to the set of values that results from evaluating the coverage with respect to each of the geometric objects that share a boundary. Appropriate values of the CV_CommonPointRule include 'all', 'average', 'high', and 'low'. For example, data used for bathymetric purposes may make use of the 'high' value (depending on the Z axis direction) to ensure that obstructions such as rocks or shoals are emphasised. In the case of discrete point coverages, *commonPointRule* is relevant only if a spatial query returns more than one point element of the coverage within the tolerance of the query.

8-7.4 S100_IF_TINCoverage spatial model

A TIN coverage is a type of CV_ContinousCoverage as described in ISO 19123. The attribute values in the value record for each CV_GeometryValuePair represent values for each of the vertex corners of the triangle. Any additional attributes related to a TIN triangle may be described as attributes of CV_ValueTriangle.

A TIN covers an area with a unique set of non-overlapping triangles where each triangle is formed by three points. The geometry for a TIN is described in ISO 19107 and a TIN coverage is described in ISO 19123. TIN coverages are particularly useful for representing elevation or bathymetry in some applications. It is easier to calculate an intersection with a coverage surface when it is represented as a TIN. The class S100_IF_TINCoverage is illustrated in Figure 8-23.

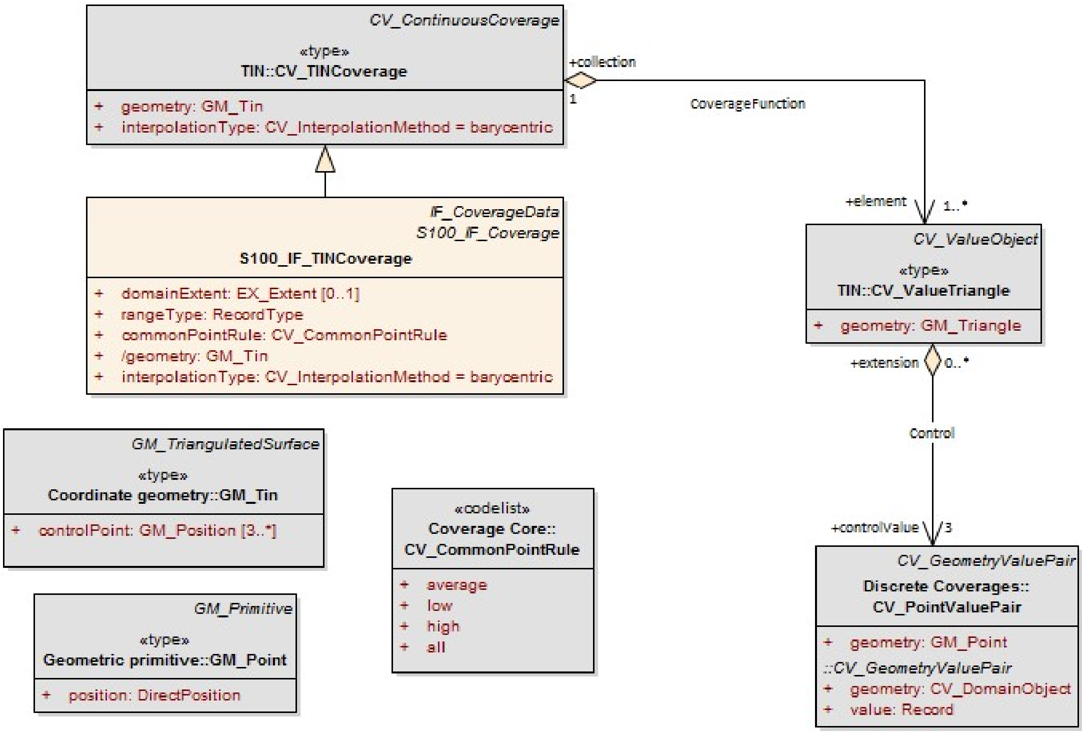


Figure 8-23 – S100_IF_TINCoverage

Table 8-8 — S100_IF_TINCoverage

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_IF_TINCoverage	Spatial type for TIN coverages	-	-	
Attribute	domainExtent	The spatiotemporal extent of the domain of the coverage	0..1	EX_Extent	Required only if the extent is different from the extent of the owning dataset See Note 1

Attribute	rangeType	Describes the range of the coverage A list of name:data type pairs each of which describes an attribute type included in the range of the coverage	1	RecordType	Implemented externally as Feature Catalogue (S-100 Part 5) and feature information group (S-100 Part 10c)
Attribute	commonPointRule	Procedure used for evaluating the coverage at a position that falls on the boundary or in an area of overlap between geometric objects in the domain of the coverage	1	CV_CommonPointRule	The ISO values “start” and “end” are not used. See Note 2
Attribute	interpolationType	The interpolation method to be used in evaluating the coverage	1	CV_InterpolationMethod	See Note 3
Attribute	geometry	Describes the network of triangles that form the basis of the TIN	1	GM_Tin	See Note 4

NOTE 1: *domainExtent* may be implemented as a geographic bounding box or bounding polygon with the temporal extent being implemented using separate attributes.

NOTE 2: The rule defined by attribute *commonPointRule* shall be applied to the set of values that results from evaluating the coverage with respect to each of the geometric objects that share a boundary. Appropriate values of the CV_CommonPointRule include ‘all’, 'average', 'high', and 'low'. For example, data used for bathymetric purposes may make use of the 'high' value (depending on the Z axis direction) to ensure that obstructions such as rocks or shoals are emphasised. The use of the *commonPointRule* occurs where a set of geometric objects are involved, such as the triangles in a TIN.

NOTE 3: The attribute *interpolationType* specifies the interpolation method recommended for the evaluation of the S100_IF_TINCoverage where the value is taken from the codelist CV_InterpolationMethod with the value "barycentric". The barycentric position S within a value triangle composed of the CV_PointValuePairs (P1, V1), (P2, V2), and (P3, V3), is (i, j, k), where S = iP1 + jP2 + kP3 and the interpolated attribute value at S is V = iV1 + jV2 +kV3.

NOTE 4: The triangles lie on a 2 dimensional manifold with the X,Y coordinates of the points at the vertices of the triangles representing the position on the manifold and the attribute. Three vertex points define a triangle. The attribute *geometry* for a TIN vertex is an instance of GM_Point. The attribute value contains a record restricted to one entry that defines the coverage value at the vertex (for example depth for a bathymetric TIN vertex point).

8-7.5 S100_IF_GridCoverage spatial model

The class S100_IF_GridCoverage (Figure 8-24 below) represents a set of values assigned to the points in a 2D grid. Several organizations of grids are available from ISO 19123 with different grid traversal orders, and variable or fixed grid cell sizes. S-100 makes use of two types of grid organizations, the simple quadrilateral grid with equal cell sizes traversed by a linear sequence rule, and the variable cell size quadrilateral grid traversed by a Morton Order sequence rule. This variable cell size grid organization is known as the Quad Tree for a two dimensional grid.

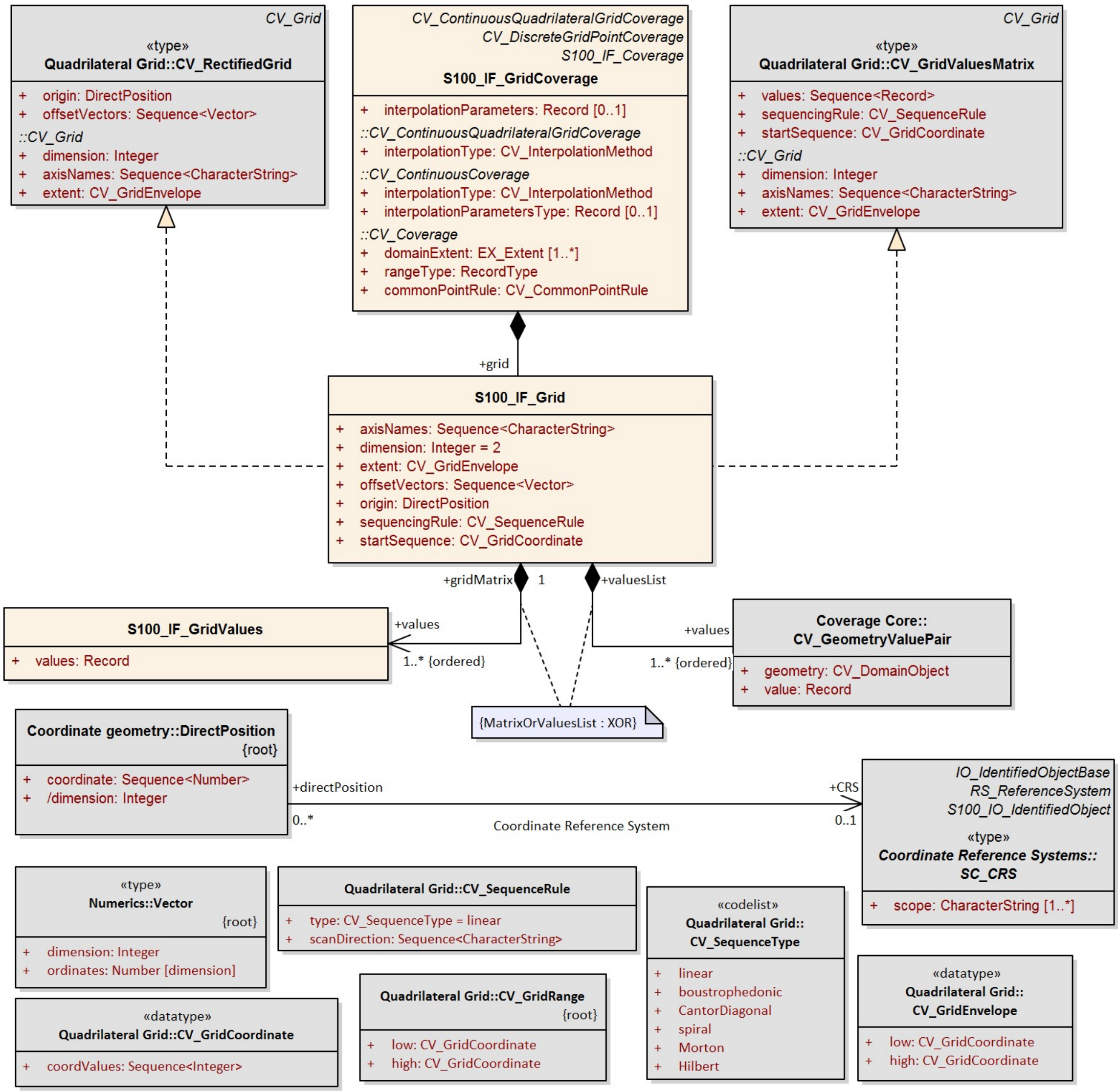


Figure 8-24 - S100_IF_GridCoverage

Table 8-9 — S100_IF_GridCoverage

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_IF_GridCoverage	Coverage type for grid coverages	-	-	Subclasses both continuous and discrete grids
Attribute	domainExtent	The spatiotemporal extent of the domain of the coverage	0..1	EX_Extent	Required only if the extent is different from the extent of the owning dataset See Note 1
Attribute	rangeType	Describes the range of the coverage A list of name:data type pairs each of which describes an attribute type included in the range of the coverage	1	RecordType	Implemented externally as Feature Catalogue (S-100 Part 5) and feature information group (S-100 Part 10c).

Attribute	commonPointRule	Procedure used for evaluating the coverage at a position that falls on the boundary or in an area of overlap between geometric objects in the domain of the coverage	1	CV_CommonPointRule	The ISO values “start” and “end” are not used See Note 2
Attribute	interpolationType	The interpolation method to be used in evaluating the coverage	1	S100_CV_InterpolationMethod	See Note 3
Attribute	interpolationParameters	Holds the values of the parameters required to execute the interpolate operation specified by interpolationType	0..1	Record	Conditional in S-100 to interpolation methods ‘biquadratic’ and ‘bicubic’ only Substitutes for ISO 19123 attribute interpolationParametersType
Composition	grid	The grid parameters, geometry, and values	1	S100_IF_Grid	A grid coverage must have grid geometry and values

NOTE 1: *domainExtent* may be implemented as a geographic bounding box or bounding polygon with the temporal extent being implemented using separate attributes. This attribute overrides the mandatory *domainExtent* inherited from CV_Coverage.

NOTE 2: The rule defined by attribute *commonPointRule* shall be applied to the set of values that results from evaluating the coverage with respect to each of the geometric objects that share a boundary. Appropriate values of the CV_CommonPointRule include ‘all’, 'average', 'high', and 'low'. For example, data used for bathymetric purposes may make use of the 'high' value (depending on the Z axis direction) to ensure that obstructions such as rocks or shoals are emphasised. The use of the *commonPointRule* occurs where a set of geometric objects are involved, such as the triangles in a TIN.

NOTE 3: The attribute *interpolationType* describes the interpolation method recommended for evaluation of the S100_IF_GridCoverage. The interpolation methods available are: Bilinear interpolation, Bicubic interpolation, Nearest-neighbour, and Biquadratic interpolation. These methods are defined in ISO 19123. Discrete point grids must use the special value for “no interpolation” added by S-100 (see S100_CV_InterpolationMethod).

The class S100_IF_Grid is a realization of CV_RectifiedGrid and CV_GridValuesMatrix from ISO 19123. The attributes inherit from the classes in ISO 19123.

Table 8-10 — S100_IF_Grid

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_IF_Grid	Type for grid geometry and values	-	-	
Attribute	axisNames	Names of the grid axes		<Sequence>CharacterString	Must be as in the EPSG Registry
Attribute	dimension	The dimension of the grid	1	Integer	Number of spatial dimensions, for example 2 for 2D grids
Attribute	extent	Area of the grid for which data are provided	1	CV_GridEnvelope	See Note 1
Attribute	offsetVectors	The spacing between grid points and the orientation of the grid axis with respect to the external coordinate reference system identified through the attribute origin	1	<Sequence>Vector	
Attribute	origin	Coordinates of the grid origin with respect to an external coordinate system	1	DirectPosition	

Attribute	sequencingRule	Method to be used to assign values from the sequence of values to the grid coordinates	1	CV_SequenceRule	See Notes 2, 3, 4
Attribute	startSequence	CV_GridCoordinate to specify the grid coordinates of the grid point to which the first in the sequence of values is to be assigned	1	CV_GridCoordinate	The choice of a valid point for the start sequence is determined by the sequencing rule
Composition	values	Sequence of Records each containing one or more values to be assigned to a single grid point	1..*	S100_IF_GridValues OR CV_GeometryValuePair	See Note 5

NOTE 1: Grid coordinates of the corner of the area having the lowest grid coordinate values and the corner having the highest grid coordinate values. CV_GridCoordinate is specified in 19123 as a sequence of Integer values which identify a grid point, there being one integer value for each dimension of the grid. The ordering shall be the same as that of the elements of *axisNames*. The value of a single coordinate shall be the number of offsets from the origin of the grid in the direction of a specific axis.

NOTE 2: For simple grids with equal cell sizes, if data is not available for the whole area within this rectangle, then padding with null values shall be used to represent areas where no data is available. For variable cell size grids (Quad Tree grids) a characteristic of the Morton Order traversal is that nonrectangular areas may be represented. In this case the attribute *extent* is a bounding rectangle that encloses the area of the grid for which data are provided.

NOTE 3: Only the values "linear" (for a simple regular cell size grid) and "Morton" (for a Quad Tree Grid) shall be used for data that conforms to this standard. The sequence rule for a regular cell size grid is simple. When the cells are all of the same size, the cell index can be derived from the position of the Record within the sequence of Records. For a variable cell size grid the sequence order is more complex. The cell index either needs to be carried with each of the associated record values or it can be calculated based on each cell size.

NOTE 4: For simple grids with equal cell sizes the *sequencingRule* attribute of an S100_IF_Grid equals "linear" and the offset vector establishes the cell size. The attribute extent specifies the area of the grid for which data is provided. For variable cell size grids (Quad Tree grids) the *sequencingRule* attribute equals Morton and the offset vector establishes the minimum cell size. The actual cell size is included as an attribute in the data record that describes the level of aggregation of the quad structure. The attribute *extent* specifies a bounding rectangle within which data is provided. Which cells are included in the dataset is determined from the Morton ordered sequence of cells.

NOTE 5: Geometry (point coordinates in the form of an ordered point set) is provided only for grid types other than regular grid or variable cell size grids.

Table 8-11 — S100_IF_GridValues

Role Name	Name	Description	Mult	Data Type	Remarks
Class	S100_IF_GridValues	Type for grid values	-	-	
Attribute	values	Ordered list of values for the feature attributes as specified in the feature catalogue	1	Record	See Notes 1 and 2

NOTE 1: Must conform to the RecordType specified by the *rangeType* attribute. If the value of an attribute is missing or unknown, a fill value must be used in its place.

NOTE 2: For simple grids with equal cell sizes the attribute values may be only data values, but for the variable cell size Quad Tree grid the record type shall include an index number and the cell size (aggregation level) for the cell.

8-7.6 Rectified or georeferenceable grids

The S100_IF_Grid model described in clause **Error! Reference source not found.** is capable of representing rectified grids as well as referenceable grids. The model given below in Figure 8-25 below shows that a Grid can be of two types Rectified or Georeferenceable; and that the GridValuesMatrix is a subtype of the general grid object. The referenceable grid type is a subclass of CV_Grid and does not add attributes. The difference between georectified and georeferenceable grids lies in the operations for coordinate conversion:

- For georectified grids, the operation to map grid coordinates to direct positions uses the values of the attributes *origin* and *offsetVectors* in an affine transformation.
- For georeferenceable grids, the operation to map grid coordinates to direct positions is not defined in terms of *origin* and *offset verctor*. It there is no predefined association between one cell's location and that of another; each cell's location might be independently calculated.

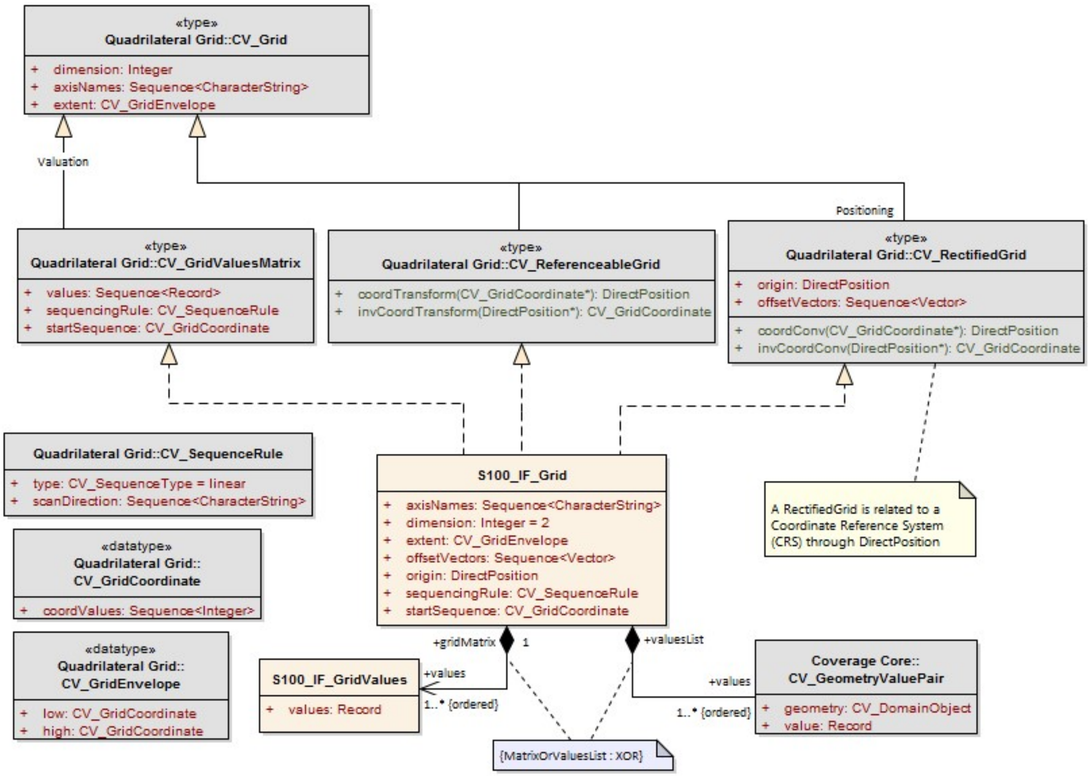


Figure 8-25 – Rectified or georeferenceable grids

A Rectified Grid is related to the Coordinate Reference System through the attribute *origin*, the orientation of the grid axes, and the spacing between the grid lines. For Rectified Grids, there is an affine relationship between the grid and the external coordinate system.

A Referenceable Grid may be related to a Coordinate Reference System through a Transform operation that allows the location of all points in the grid to be determined in the Coordinate Reference System, but the location of the points is not directly available from the grid coordinates; as opposed to a rectified grid where the location of the points in the Coordinate Reference System is derivable from the properties of the grid itself.

The S-100 model does not use coordinate transformations for georeferenceable grids. Instead, the Direct Positions of the grid points are encoded along with the value record for each grid point.

8-7.7 Common enumerations

8-7.7.1 CV_CommonPointRule

ISO 19123 states that "CV_CommonPointRule is a list of codes that identify methods for handling cases where the DirectPosition input to the evaluate operation falls within two or more of the geometric objects. The interpretation of these rules differs between discrete and continuous coverages. In the case of a discrete coverage, each CV_GeometryValuePair provides one value for each attribute. The rule is applied to the set of values associated with the set of CV_GeometryValuePairs that contain the DirectPosition. In the case of a continuous coverage, a value for each attribute shall be interpolated for each CV_ValueObject that contains the DirectPosition. The rule shall then be applied to the set of interpolated values for each attribute."

Table 8-12 — CV_CommonPointRule enumeration

Item	Name	Description	Code	Remarks
Enumeration	CV_CommonPointRule	Codes that identify methods for evaluating the coverage at positions that fall on the boundary or in an area of overlap between geometric objects in the domain of the coverage	-	ISO 19123 CV_CommonPointRule
Literal	average	Return the mean of the attribute values	1	
Literal	low	Use the least of the attribute values	2	
Literal	high	Use the greatest of the attribute values	3	
Literal	all	Return all the attribute values that can be determined for the position	4	

NOTE: Use of 'start' and 'end' is prohibited for Product Specifications conforming to this edition of S-100, since segmented curves are not included among the coverages defined in Part 8 of this Edition.

8-7.7.2 CV_SequenceType

The scan methods are described in detail in ISO 19123. The order in which scanning takes place is the same as the order of axes in the attribute *scanDirection*. The starting location of the scan is given in the attribute *startSequence*.

NOTE: Product Specification authors and producers should take care that the start location is compatible with the sequence rule and scan direction; for example, linear sequencing would be incompatible with a start location at the upper bound of the grid bounding box and forward scan order in *scanDirection*.

Table 8-13 — CV_SequenceType enumeration

Item	Name	Description	Code	Remarks
Enumeration	CV_SequenceType	Codes that identify the method of ordering grid points or value records	-	ISO 19123 CV_ SequenceType
Literal	linear	Sequencing is consecutive along grid lines, starting with the first grid axis listed in scanDirection	1	For example, for 2-D grids with scan direction=(x,y), scanning will be in row-major order
Literal	boustrophedonic	Variant of linear sequencing in which the direction of the scan is reversed on alternating grid lines. For grids of dimension > 2, it is also reversed on alternating planes	2	
Literal	CantorDiagonal	Sequencing in alternating directions along parallel diagonals of the grid. For dimension > 2, it is repeated in successive planes	3	
Literal	spiral	Sequencing in spiral order	4	
Literal	Morton	Sequencing along a Morton curve	5	
Literal	Hilbert	Sequencing along a Hilbert curve	6	

Morton curves are generated by converting the grid coordinates (axial indexes) of each grid point to binary numbers and interleaving the binary digits of the results to produce the Morton code of the grid point. The method is documented in computer science textbooks as well as ISO 19123 and other accessible articles. Hilbert curves are more complex but descriptions are available in computer science and other reference texts (for example, the non-normative references in S-100 Part 10c).

8-7.7.3 S100_CV_InterpolationMethod

S100_CV_InterpolationMethod extends the ISO 19123 codelist CV_InterpolationMethod with the 'discrete' literal. The ISO 19123 CodeList CV_InterpolationMethod includes nine interpolation methods. Each is used in the context of specified grid types, indicated in the Remarks column. S-100 adds a 'discrete' literal for use when there is no interpolation.

Table 8-14 — S100_CV_InterpolationMethod enumeration

Item	Name	Description	Code	Remarks
Enumeration	S100_CV_InterpolationMethod	Codes for interpolation methods between known feature attribute values associated with geometric objects in the domain of the discrete coverage	-	Extension of ISO 19123 CV_ InterpolationMethod
Literal	nearestneighbor	Assign the feature attribute value associated with the nearest domain object in the domain of the coverage	1	Any type of coverage
Literal	bilinear	Assign a value computed by using a bilinear function of position within the grid cell	5	Only quadrilateral grids
Literal	biquadratic	Assign a value computed by using a biquadratic function of position within the grid cell	6	Only quadrilateral grids
Literal	bicubic	Assign a value computed by using a bicubic function of position within the grid cell	7	Only quadrilateral grids
Literal	barycentric	Assign a value computed by using the barycentric method described in ISO 19123	9	Only TIN
Literal	discrete	No interpolation method applies to the coverage	10	Value added by S-100 to CV_InterpolationMethod

NOTE 1: The literals *linear*, *quadratic*, and *cubic* are prohibited since this Edition of S-100 does not include segmented curve coverages. The *lostarea* method is also omitted since this applies to Thiessen polygons which are not used in this Edition of S-100.

NOTE 2: Interpolation parameters, if needed, must be encoded in the *interpolationParameters* (see Table 8-9 in clause 8-7.5).

8-8 Data Spatial Referencing

Spatial referencing for gridded data and for point set data and TIN data are handled differently. Point set data includes a coordinate direct position for each point in the point set. TIN data includes a point at each vertex of a TIN triangle. Spatial referencing of direct positions is described in ISO 19111 Spatial referencing by coordinates, and is the same for point set, and TIN data as it is for other types of vector data. Gridded data references the grid as a whole.

8-8.1 Gridded data spatial referencing

The two spatial properties of gridded data describe how the spatial extent was tessellated into small units and spatial referencing to the earth. The ISO 19123 standard indicates that a grid may be defined in terms of a coordinate reference system. This requires additional information about the location of the grid's origin within the coordinate reference system, the orientation of the grid axes, and a measure of the spacing between the grid lines. A grid defined in this way is called a rectified grid. If the coordinate reference system is related to the Earth by a datum, the grid is a georectified grid. The essential point is that the transformation of grid coordinates to coordinates of the external coordinate reference system is an affine transformation. The class SC_CRS is specified in ISO 19111. A referenceable grid is one that can be converted to a rectified grid by a coordinate transform.

8-8.1.1 Georectified

Georectified gridded data is *uniformly spaced* gridded data. Any cell in a georectified gridded data can be uniquely geolocated, given the cell spacing, grid origin and orientation. In most georectified gridded data, cell size is constant across the whole coverage and also equates to the cell spacing. (Note, however, that uniformly spaced gridded data may be uniformly spaced in terms of image coordinates, and not geolocatable.) For georectified gridded data, information as simple as the map coordinate values of any two cells not in the same row and column can geolocate all cells in the coverage to the map coordinate system, since cell spacing, grid origin and orientation can be derived from the coordinates of the two cells.

It should be pointed out that the cell spacing (that is, cell size) in the above definition is the distance measured at the map projection coordinate system. Uniform spacing in a map coordinate system may not necessarily indicate equal spacing on the earth's surface, depending on the map projection selected. For example, a cell size of 0.1 degree longitude in the geographic coordinate system (that is lat/long) corresponds to different surface distances in kilometres at high and low latitudes.

The term "uniform spacing" means that there is equal spacing in some defined coordinate system. "Regular spacing" means that there is some function that equates location to cell spacing.

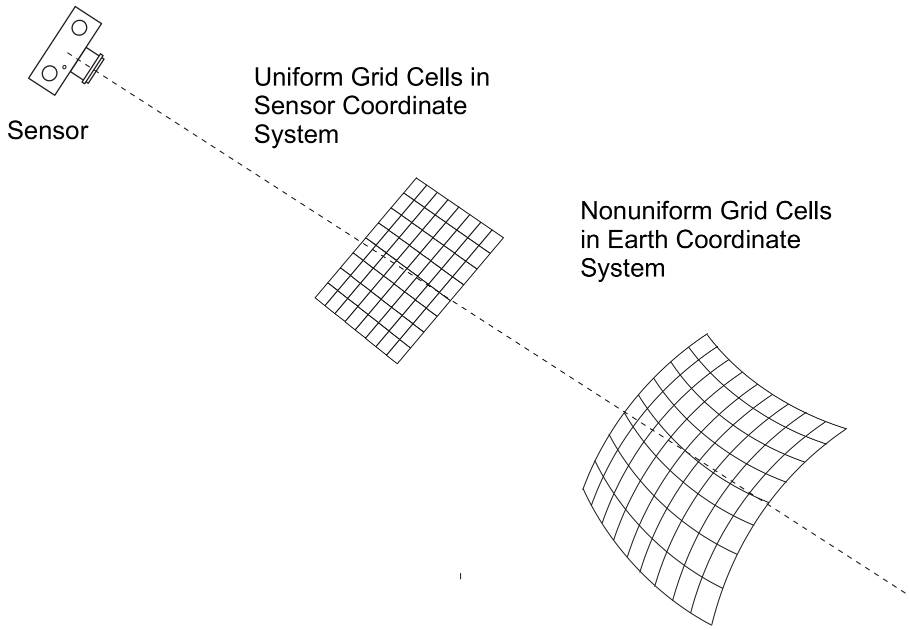


Figure 8-26 – Non uniform spacing of grid cells

8-8.1.2 Ungeorectified

Ungeorectified gridded data is geospatial gridded data whose cells are non-uniformly spaced in any geographic/map projection coordinate system. Therefore, the location of one cell in an ungeorectified gridded data cannot be determined based on another cell's location.

Ungeorectified gridded data can be further classified into *georeferenced* and *georeferenceable* subclasses, depending on whether information is provided with a dataset that allows determination of the geolocation of a cell.

8-8.1.3 Georeferenced

Georeferenced gridded data is gridded data whose cell locations can be uniquely determined through certain geolocating algorithms, such as warping, using information provided with the data. Most raw remote sensing data and raw hydrographic sonar data are in the georeferencaeble form.

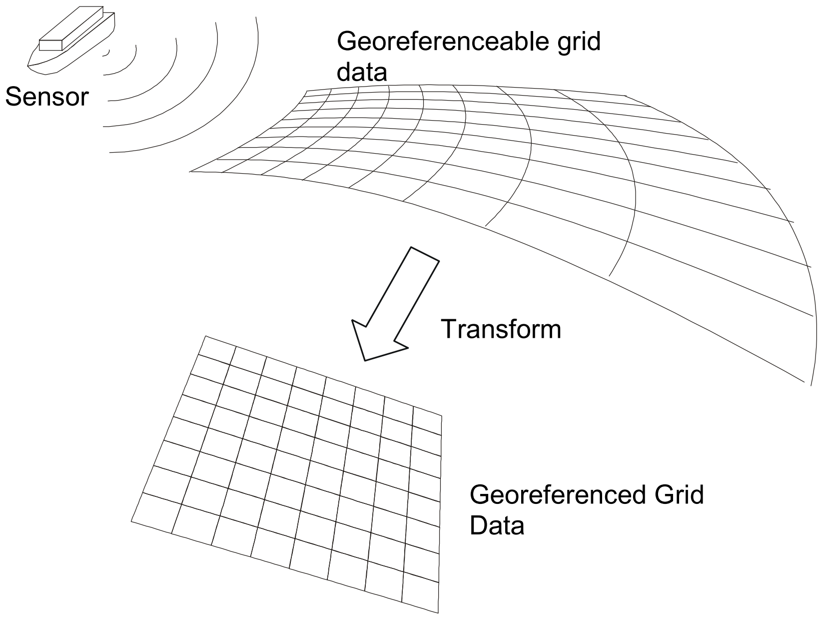


Figure 8-27 – Georectified data

8-8.1.4 Georeferenceable

Georeferenceable gridded data is ungeorectified gridded data that does not include any information that can be used to determine a cell's geographic coordinate values, for example, a digital perspective aerial photograph without georectifiction information included. (An aerial photograph can be georeferenced through a set of ground control points.)

The difference between georectified and georeferenced data is that cell spacing is constant in a georectified data while it may be variable in georeferenced data. In georectified data, the location of any cell can be determined given the data's cell spacing, grid orientation and the coordinates of any one cell. In georeferenced data, there is no predefined association between one cell's location and that of another; each cell's location might be independently calculated. Georectified gridded data are normally obtained from georeferenced data through georectification (also called geometric correction). The georectification process involves two steps. The first step is to calculate the grid coordinates (for example row and column) of regularly spaced cells located at the map coordinate x, y. This step is called *coordinate mappin*g. The second step is to assign the cell with an attribute value based on the attribute values at the corresponding and neighbouring grid coordinates. This step is called *resamplin*g. Spatial referencing information for imagery data is carried as metadata.

8-8.2 Point set data and TIN triangle vertex spatial referencing

Point sets and TIN triangles are described in the ISO Spatial Schema standard 19107, which has been profiled as part of S-100. Each point in a point set is located by a direct position. The spatial referencing system that relates to the direct positions in the set is referenced by the Spatial Schema, through the same SC_CRS object.

8-9 Imagery and Gridded Data Metadata

The general structure for imagery and gridded data given in Figure 8-3 shows that metadata is one of the primary components of an imagery and gridded dataset. A gridded dataset consists of attribute data contained in a grid value matrix and associated metadata. Everything except for the actual grid cell attributes is metadata. Some of the metadata is structural, such as the metadata required to define the geometric structure or spatial referencing, while other metadata describes the meaning of the dataset. Some of the structural metadata will be carried as attributes of the Grid Value Matrix Object. Figure 8-28 below is a model showing the relationship to metadata for all coverage data.

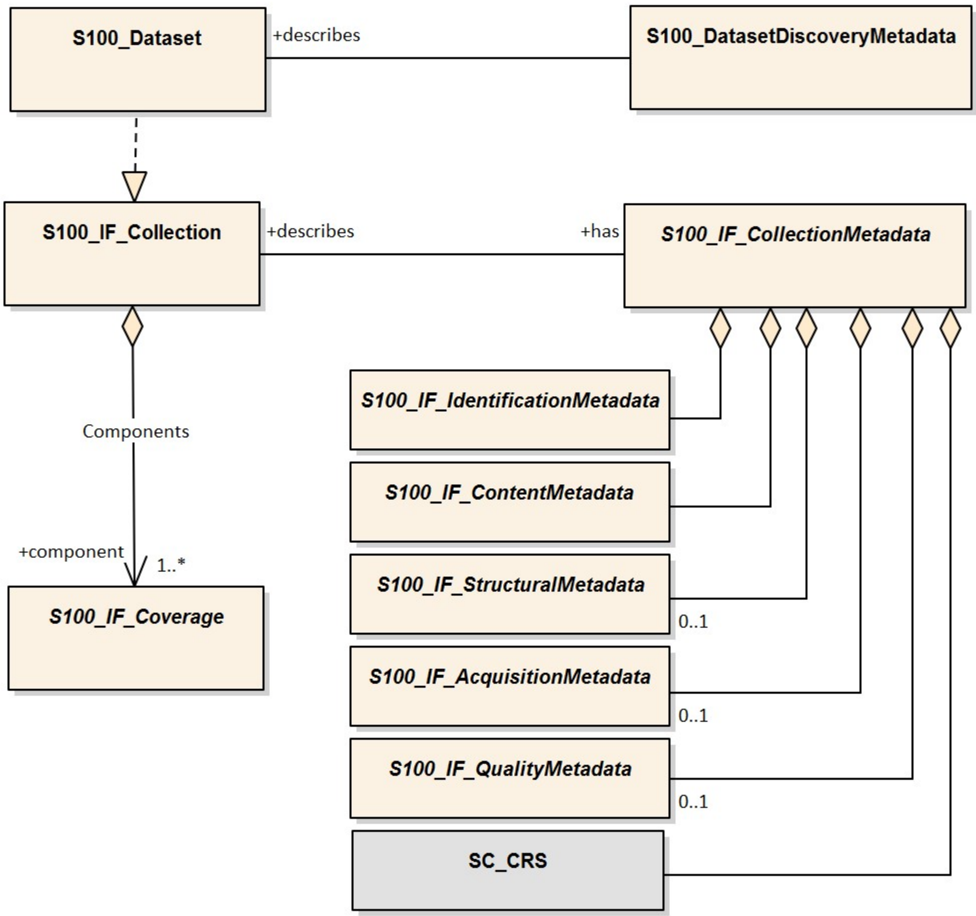


Figure 8-28 – Relationship to metadata

The metadata for all types of geographic data is covered in the metadata standard ISO 19115-1 Metadata. This standard includes mandatory identification metadata that describes the data set. This is called Catalog or Discovery metadata. It also includes some metadata describing the content of a data set. This is particularly true at the feature level. Much of the metadata corresponding to vector based geometric data does not apply to imagery and gridded data. The metadata elements in 19115-1 are used where possible to address the requirements for Imagery, Gridded and Coverage data. Some basic imagery metadata elements are already defined in 19115-1. Other metadata elements primarily related to acquisition and processing are addressed in ISO 19115-2 Geographic information - Metadata - Part 2: Extensions for imagery and gridded data. The minimum amount of metadata required to describe a coverage data is addressed in 19115-1. The details of sensor models and their associated data models and metadata are provided in ISO 19130 Geographic information - Sensor and data models for imagery and gridded data. Metadata for S-100 is given in S-100 Part 4. The specific metadata for S-100 Imagery and Gridded Data is shown in Appendix 8-D.

8-10 Quality

The general concept for handling quality in the ISO 19100 series of standards is defined in the ISO 19113 “Quality principles”. The procedures to evaluate quality are defined in the ISO 19114 “Quality evaluation procedures”. ISO 19138:2006 “Data quality measures” provides a definitive set of measures. The metadata quality elements from ISO 19115:2003 have been moved to a new standard ISO 19157:2013 Geographic information -- Data quality.

The ISO 19129 standardizes quality aspects that are specific for imagery, gridded, and coverage data. The testing of the quality according to this standard is model based. The quality measures are attributes or constraints of the classes of the model. Appendix 8-C shows the proposed top-level classes of the quality model.

8-11 Imagery and Gridded Data Portrayal

The mechanism for portrayal is out of scope for this component of S-100. It is described in the Portrayal component of S-100 Parts 9 and 9a. The basic mechanism for feature centric rule based portrayal is given in ISO 19117 "Portrayal". However, certain information may need to be carried with a set of imagery and gridded data to support external portrayal mechanisms.

8-12 Imagery and Gridded Data Encoding

Details of encoding are out of scope of this component of S-100. The standard S-100 encoding for coverage datasets is described in S-100 Part 10c. An S-100 standard for "picture/image" encoding has not been defined as yet, but the References clause in this Part and the list of allowed support file formats in S-100 Part 17 may be consulted for hints on formats which may be of use.

8-13 Metadata for Scanned Image

This clause suggests associated metadata for the use of supporting a scanned paper chart in compliance with S-61.

The following Table assigns the metadata identified in S-61 to the metadata classes in ISO 19115-1 and ISO 1915-2.

Table 8-15 — S-61 Metadata in terms of ISO 19115-1 and ISO 19115-2

S-61	ISO 19115-1/2 class
Producing Agency	MD_Metadata - contact - CI_Responsibility (including organization name, contact info and role of producing agency)
	MD_Metadata - identificationInfo - MD_Identification - purpose - "Raster Nautical Chart"
	MD_Constraints_useLimitation
	MD_Constraints_MD_LegalConstraints
RNC number	MD_Identification - citation - CI_Citation - identifier
Chart identifier	LI_Lineage - LI_Source - sourceCitation - CI_Citation - identifier
RNC edition date	MD_Metadata - dateStamp - Date
Chart edition date	LI_Lineage - LI_Source - sourceCitation - CI_Citation - edition
Last update or Notice to Mariner applied	LI_Lineage - LI_Source - SourceStep - LI_ProcessStep_dateTime
	MD_DataIdentification - topicCategory - TopicCategoryCode
	MD_DataIdentification - SpatialRepresentationType - SpatialRepresentationTypeCode - "2" (grid)
Chart scale	MD_ReferenceSystem
Orientation of North	MD_ReferenceSystem
Projection and projection parameters	MD_ReferenceSystem
Horizontal Datum	MD_ReferenceSystem
Horizontal Datum shift	MD_ReferenceSystem
Vertical datums	MD_ReferenceSystem
Depth and Height units	MD_ReferenceSystem or MD_Identification – EX_Extent – EX_VerticalExtent –

S-61	ISO 19115-1/2 class
	MD_ReferenceSystem or MD_Identification – EX_Extent – EX_VerticalExtent – SC_VerticalCRS – axisUnitID: unitOfMeasure
Pixel Resolution	MD_DataIdentification - spatialResolution - MD_Resolution -
Transform to allow geographic positions to be converted to RNC coordinates	MD_ReferenceSystem
Colour palettes for daytime, nighttime and dusk	MD_PortrayalCatalogueReference
Information to handle notes, diagrams and marginalia	Notes and textual marginalia may be captured as MD_MetadataExtensionInformation, whereas diagrams must be handled by reference to an associated data file containing the diagram.
Source diagram	Textual description of source may be captured as MD_MetadataExtensionInformation, whereas a source diagram must be handled by reference to an associated data file containing the diagram.
Update metadata including: - producer of update; - update number; - date; - identifier of which RNC to which it applies; - chart edition to which it applies; - changes to metadata; and - information so it can be applied automatically	MD_MaintenanceInformation together with MD_Identification

8-14 Feature Oriented Image

All gridded datasets are feature oriented, in that a coverage is a subtype of a feature. That is an entire gridded dataset can be considered to be a single feature. A feature structure can be applied to gridded data in two different ways. First, a discrete coverage can carry a feature code as an attribute. For example, a coverage corresponding to the postal code system will have discrete values for each postal code, yet still cover the country completely. The only difference in the Application Schema is a relationship between the discrete coverage and the feature. This is shown in Figure 8-29.

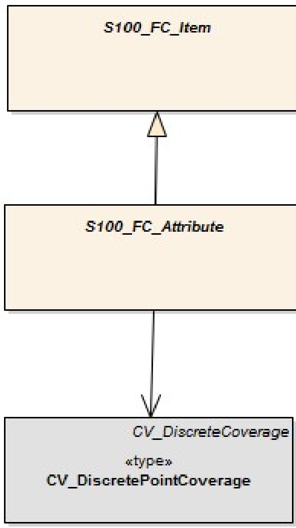


Figure 8-29 – Feature oriented discrete coverage

The second method of establishing a feature structure is to develop a composite dataset that contains many separate but adjoining coverages. The coverages may be continuous or discrete. This is very much like the way a "vector" data set is composed where each feature has its own geometry and attributes. In fact vector data may be mixed with coverage data in the same dataset. The Application Schema simply allows multiple instances of features.

Geometric elements such as grids may be shared between multiple features, and features may be related by composition or other relationships as allowed in the general feature model of ISO 19109. A complex feature may include both a continuous grid coverage and vector data such as a polygonal boundary. A feature oriented dataset may contain both a continuous coverage of the ocean as collected by sonar, and point and line features corresponding to navigational aids. Topological primitives may relate all of the features. This allows for some interesting and useful structures.

A Raster Nautical Chart may include additional vector data describing the navigational aids, hazards and danger zones, which is not "visible" in that it is not portrayed, but which is active in the use of the Raster Nautical Chart, so a ship can determine whether it is within a danger zone, or perform some other ECDIS-like functions.

See Appendix 8-E for additional information about Feature Oriented Gridded Data.

Page intentionally left blank

Appendix 8-A
Abstract Test Suite
(normative)

8-A-1 Quadrilateral Grid

1) Test Purpose: Verify that an Application Schema instantiates the classes defined in ISO 19123 of CV_Grid, CV_GridPoint, CV_GridCell, CV_GridValuesMatrix, CV_GridPointValuePair, CV_DiscreteGridPointCoverage, or CV_ContinuousGridCoverage, and CV_GridValueCell with their specified attributes, operations, associations and constraints, in the context of the classes S100_IF_GridCoverage, S100_IF_Grid and S100_IF_GridValues as defined in this Standard
2) Test Method: Inspect the documentation of the Application Schema or profile.
3) Reference: ISO 19123, Clause 8.
4) Test Type: Capability.

8-A-2 Scanned Image

1) Test Purpose: Verify that an Application Schema for Raster Scan Image satisfies the requirements of A.1; that it includes the metadata elements identified in Table 8-15.
2) Test Method: Inspect the documentation of the Application Schema or profile.
3) Reference: ISO 19115-1, IHO S-61.
4) Test Type: Capability.

8-A-3 TIN Coverage

1) Test Purpose: Verify that an Application Schema for TIN Coverage instantiates the classes defined in ISO 19123 of CV_TINCoverage, CV_ValueTriangle, and CV_GridPointValuePair with their specified attributes, operations, associations and constraints, in the context of the class S100_IF_TINCoverage as defined in this Standard.
2) Test Method: Inspect the documentation of the Application Schema or profile.
3) Reference: ISO 19123
4) Test Type: Capability.

8-A-4 Point Coverage

1) Test Purpose: Verify that an Application Schema for Point Coverage instantiates the classes defined in ISO 19123 of CV_DiscretePointCoverage, and CV_PointValuePair, with their specified attributes, operations, associations and constraints, in the context of the class S100_IF_PointCoverage as defined in this Standard.
2) Test Method: Inspect the documentation of the Application Schema or profile.
3) Reference: ISO 19123
4) Test Type: Capability.

8-A-5 Point Set

1) Test Purpose: Verify that an Application Schema for Point Set instantiates the classes defined in ISO 19107 of GM_Point, with its specified attributes, operations, associations and

constraints, in the context of the classes S100_IF_PointSet and S100_Point as defined in this Standard.

2) Test Method: Inspect the documentation of the Application Schema or profile.
3) Reference: ISO 19107
4) Test Type: Capability.

8-A-6 Variable Cell Size Grid

1) Test Purpose: Verify that an Application Schema for Variable Cell Size instantiates the classes defined in ISO 19123 of CV_Grid, CV_GridPoint, CV_GridCell, CV_GridValuesMatrix, CV_GridPointValuePair, CV_DiscreteGridPointCoverage, or CV_ContinuousGridCoverage, and CV_GridValueCell with their specified attributes, operations, associations and constraints, with the CV_ContinuousCoverage CV_InterpolationMethod attribute set to NearestNeighbour and the CV_GridValuesMatrix CV_SequenceRule attribute set to (x,y) Morton.
2) Test Method: Inspect the documentation of the Application Schema or profile.
3) Reference: ISO 19123
4) Test Type: Capability.

8-A-7 Feature Oriented Image Discrete Coverage

1) Test Purpose: Verify that an Application Schema for Feature Oriented Image that uses a discrete coverage instantiates the classes defined in ISO 19123 of CV_Grid, CV_GridPoint, CV_GridCell, CV_GridValuesMatrix, CV_GridPointValuePair, CV_DiscreteGridPointCoverage, CV_DiscreteCoverage, and CV_GeometryValuePair with their specified attributes, operations, associations and constraints.
2) Test Method: Inspect the documentation of the Application Schema or profile.
3) Reference: ISO 19123, 19109
4) Test Type: Capability.

8-A-8 Feature Oriented Image in a Multi-feature Environment

1) Test Purpose: Verify that an Application Schema instantiates the classes defined in ISO 19123 of CV_Grid, CV_GridPoint, CV_GridCell, CV_GridValuesMatrix, CV_GridPointValuePair, CV_DiscreteGridPointCoverage, or CV_ContinuousGridCoverage, and CV_GridValueCell with their specified attributes, operations, associations and constraints, and that multiple features are permitted with separate CV_Coverages or GM_Objects.
2) Test Method: Inspect the documentation of the Application Schema or profile.
3) Reference: ISO 19123, 19109, 19107
4) Test Type: Capability

Appendix 8-B
Terminology
(informative)

The terminology used in S-100 aligns with the terminology used in the ISO 19100 suite of standards and it is different from that used in S-57. S-57 uses the terms “raster” and “matrix” to address images and data described by organized sets of attribute values. The ISO 19100 suite of standards has a more rigorous definition of terms, but these new terms include much more that is normally thought of as "Raster" or "Matrix" data. Unfortunately current terms in this field have been used with broad overlapping meanings and the terminology can be confusing.

One of the most misused terms is “raster”. Technically the term describes the row by column scanning of a regular rectangular grid, such as the raster scan of a television screen. A raster is a type of a grid. However, often the term is used in a very broad sense to mean most, but not all types of data that cover an area. S-100 now makes use of the term “raster” in its more precise technical sense as a traversal method for a grid of data.

“matrix” is a term that is also used in different ways in different contexts. It is sometimes colloquially used to address all gridded data that corresponds to measurements from non-imaging sensors. But what is an imaging sensor? What is an image? Anything that can be "seen" is thought of as being an image. But a graph of measured data such as elevations, even a two-dimensional graph of data, can be seen. In fact visualization is the purpose of graphing. The term “matrix” also has a mathematical meaning of being an organized set of numbers. The current colloquial meaning of the term “matrix” has been abandoned in this edition of S-100, and the mathematical meaning of an ordered set of numbers is retained as the meaning for the word.

ISO begins defining its terminology by defining a "coverage". In TC211, a coverage is defined as a "function to return one or more feature attribute values for any direct position within its spatiotemporal domain". For a continuous coverage any position in the spatiotemporal domain has a value. A coverage function is basically an interpolation function over a set of grid points or other points covering an area. This makes a coverage the inverse of what is normally thought of as a set of gridded data. Data collected from a sensor creates a values matrix that drives the coverage function. This set of values may be organized in several ways. The simplest is a regular grid, but there may be other organizations of grids such as tiled grids or irregular shaped grids. There may even be grids with variable size cells in multi-dimensions that have been shown to be quite effective in handling hydrographic sounding data. The ISO 19123 standard defines a Grid Value Matrix, TIN Value Triangle, Segmented-Curve Value Curves, and Thiessen Value Polygons as the base elements for the set of data sampled from a sensor. This component of S-100 only needs the concept of a grid value matrix, and does not need to address Segmented Curves, or Thiessen Polygons.

The terms Imagery, Gridded and Coverage data are not mutually exclusive terms. Imagery is a type of Gridded data and Gridded data is a type of Coverage data. Coverage is the broad term. Grid describes one organization of the matrix of data supporting a coverage function. An image is data that may be "viewed".

S-100 needs to use terminology in alignment with ISO and other external standards. However it also needs to recognize the uses of terms in previous editions of S-57. A raster is a grid traversal method. Therefore “Raster Image Data” means data organized as a set of grid value matrix points representing an image. “Raster Image Data” corresponds generally to the term Raster Data as used in S-57 edition 3. Gridded data is all data organized as a set of grid value matrix points. Therefore “Gridded Data” corresponds generally to the term Matrix Data as used in S-57 Edition 3.

Page intentionally left blank

Appendix 8-C
Quality Model for Imagery and Gridded Data
(informative)

The following is a list of quality elements test procedures that addresses imagery. Quality elements for gridded and coverage data are described in S-100 Part 4c and S-97 Part C.

8-C-1 Top-level classes of the quality-model

General image quality
Visual inspection and evaluation of image geometry
Analytical inspection and evaluation of image geometry
Visual inspection and evaluation of image radiometry
Analytical inspection and evaluation of image radiometry

The following listings are non-exhaustive listings of the subclasses of the quality model.

8-C-2 Class General image quality

check parameters affecting the quality (data compression etc.)
make test scanning or test imaging

8-C-3 Class Visual inspection and evaluation of image geometry

check number of channels (black&white, colour, multispectral, etc.)
check edge-matching
check event of blurring
check rectification errors
check “pixel-stretching”
check overlay with vector data (other mapping data, map-frame)
check overlay with other raster or gridded data
identify source of data
inspect documentation of the quality of the sensor or the scanner (calibration data)
inspect documentation of previous processing step (image enhancements)
check resolution of imaged test patterns

8-C-4 Class Analytical inspection and evaluation of image geometry

check seam lines of mosaics
check colour stability / homogeneity / balance
check grade of illumination of the image (hot spot)
check histogram
check coloured fringes along lines with high contrast

8-C-5 Class Visual inspection and evaluation of image radiometry

calculate geometric residuals at checkpoints in 2D and/or in 3D
calculate residuals in range at checkpoints

8-C-6 Class Analytical inspection and evaluation of image radiometry

calculate contrast
calculate brightness

Page intentionally left blank

Appendix 8-D
Metadata
(informative)

Metadata for S-100 is taken where possible from the ISO 19115-1 Metadata standard to ensure a high level of compatibility with other standards based on the same metadata standard. This metadata has been organized into a number of packages. The following is a list of the packages defined in ISO 19115-1.

Relationship between packages of metadata and metadata classes

Package	Class
Metadata information	MD_Metadata
Identification information	MD_Identification
Constraint information	MD_Constraints
Data quality information	DQ_DataQuality (ISO 19157)
Maintenance information	MD_MaintenanceInformation
Spatial representation information	MD_SpatialRepresentation
Reference system information	MD_ReferenceSystem
Content information	MD_ContentInformation
Portrayal Catalogue information	MD_PortrayalCatalogueReference
Distribution information	MD_Distribution
Metadata extension information	MD_MetadataExtensionInformation
Application Schema information	MD_ApplicationSchemaInformation
Extent information	EX_Extent
Citation and responsible party information	CI_Citation CI_Responsibility

ISO TC211 has also completed ISO 19115-2 Geographic information - Metadata - Part 2: Extensions for imagery and gridded data. It contains additional packages for MI_AcquisitionInformation, Lineage (Source and Process), Quality result for Coverage (QE_CoverageDescription) and usability (QE_Usability) that are relevant for the description of Imagery and Gridded data in S100.

The MI_AcquisitionInformation package provides details specific to the acquisition of imagery and gridded data. It contains:

1) MI_Instrument, designations of the measuring instruments used to acquire the data;
2) MI_Operation, designations of the overall data gathering program to which the data contribute;
3) MI_Platform, designations of the platform from which the data were taken;
4) MI_Objective, the characteristics and geometry of the intended object to be observed;
5) MI_Requirement, the user requirements used to derive the acquisition plan;
6) MI_Plan, the acquisition plan that was implemented to acquire the data;
7) MI_Event, describes a significant event that occurred during data acquisition. An event can be associated with an operation, objective, or platform pass; and
8) MI_PlatformPass, identifies a particular pass made by the platform during data acquisition. A platform pass is used to provide supporting identifying information for an event and for data acquisition of a particular objective.

The additional classes to address the sources and production processes of particular importance for imagery and gridded data are:

1) QE_CoverageResult is a specified subclass of DQ_Result and aggregates information required to report data quality for a coverage;
2) QE_Usability is a specified subclass of DQ_Element used to provide user specific quality information about a dataset's suitability for a particular application;
3) LE_ProcessStep is a specified subclass of LI_ProcessStep and contains additional information on the history of the algorithms used and processing performed to produce the data. It includes a description of:
   a) LE_Processing, which describes the procedure by which the algorithm was applied to generate the data from the source data;
   b) LE_ProcessStepReport which identifies external information describing the processing of the data;
   c) LE_Source,which describes the output of a process step.

8-D-1 Metadata class information (MD_Metadata) from ISO 19115-1 and ISO 19157

The MD_Metadata class is an aggregate of the following classes (which are further explained in the following subclauses):

8-D-1.1 Identification information (MD_Identification)

Identification information contains information to uniquely identify the data. It includes information about the citation for the resource, an abstract, the purpose, credit, the status and points of contact. The MD_Identification entity is mandatory. It contains mandatory, conditional, and optional elements. MD_Identification is an aggregate of the following entities:

1) MD_Format, format of the data;
2) MD_BrowseGraphic, graphic overview of the data;
3) MD_Usage, specific uses of the data;
4) MD_Constraints, constraints placed on the resource;
5) MD_Keywords, keywords describing the resource; and
6) MD_MaintenanceInformation, how often the data is scheduled to be updated and the scope of the update.

8-D-1.2 Constraint information (MD_Constraints)

This package contains information concerning the restrictions placed on data. The MD_Constraints entity is optional and may be specified as MD_LegalConstraints and/or MD_SecurityConstraints. The otherConstraint element of MD_LegalConstraints shall be non-zero (used) only if accessConstraints and/or useConstraints elements have a value of "otherRestrictions", which is found in the MD_RestrictionCode enumeration.

8-D-1.3 Data quality information (DQ_DataQuality – ISO 19157)

This package contains a general assessment of the quality of the dataset. The DQ_DataQuality entity is optional and contains the scope of the quality assessment. DQ_DataQuality is an aggregate of LI_Lineage and DQ_Element. DQ_Element can be specified as DQ_Completeness, DQ_LogicalConsistency, DQ_PositionalAccuracy, DQ_ThematicAccuracy and DQ_TemporalAccuracy. Those five entities represent Elements of data quality and can be further subclassed to the sub-Elements of data quality. Users may add additional elements and sub-elements of data quality by sub-classing DQ_Element or the appropriate sub-element.

This package also contains information about the sources and production processes used in producing a dataset. The LI_Lineage entity is optional and contains a statement about the lineage. LI_Lineage is an aggregate of LI_ProcessStep and LI_Source. The "report" and "lineage" roles of DQ_DataQuality are mandatory if DQ_DataQuality.scope.DQ_Scope.level has a value of "dataset". The "levelDescription" element of DQ_Scope is mandatory if the "level" element of DQ_Scope does not have a value of "dataset" or "series". The "statement" element of LI_Lineage is mandatory if DQ_DataQuality.scope.DQ_Scope.level has a value of "dataset" or "series" and the LI_Lineage roles of "source" and "processStep" are not documented.

The “source” role of LI_Lineage is mandatory if the “statement” element and the “processStep” role of LI_Lineage are not documented. The “processStep” role of LI_Lineage is mandatory if the “statement” element and the “source” role of LI_Lineage are not documented. Either the “description” or “sourceExtent” element of LI_Source must be documented.

8-D-1.4 Maintenance information (MD_MaintenanceInformation)

This package contains information about the scope and frequency of updating data. The MD_MaintenanceInformation entity is optional and contains mandatory and optional metadata elements.

8-D-1.5 Spatial representation information (MD_SpatialRepresentation)

This package contains information concerning the mechanisms used to represent spatial information in a dataset. The MD_SpatialRepresentation entity is optional and can be specified as MD_GridSpatialRepresentation and MD_VectorSpatialRepresentation. Each of the specified entities contains mandatory and optional metadata elements. When further description is necessary, MD_GridSpatialRepresentation may be specified as MD_Georectified and/or MD_Georeferenceable. Metadata for Spatial data representation are derived from ISO 19107.

8-D-1.6 Reference system information (MD_ReferenceSystem)

This package contains the description of the spatial and temporal reference system(s) used in a dataset. MD_ReferenceSystem contains an element to identify the reference system used. MD_ReferenceSystem may be subclassed as MD_CRS, which is an aggregate of MD_ProjectionParameters and MD_EllipsoidParameters. MD_ProjectionParameters is an aggregate of MD_ObliqueLineAzimuth and MD_ObliqueLinePoint. MD_ReferenceSystem is derived from RS_ReferenceSystem, which can be specified as SC_CRS, SI_SpatialReferenceSystemUsingGeographicIdentifiers and TM_ReferenceSystem. Metadata for Reference system information are derived from ISO 19108, ISO 19111 and ISO 19112.

8-D-1.7 Content information (MD_ContentInformation)

This package contains information identifying the Feature Catalogue used (MD_FeatureCatalogueDescription) and/or information describing the content of a coverage dataset (MD_CoverageDescription). Both description entities are subclasses of the MD_ContentInformation entity. MD_CoverageDescription may be subclassed as MD_ImageDescription, and is an aggregate of MD_RangeDimension. MD_RangeDimension may additionally be subclassed as MD_Band.

8-D-1.8 Portrayal Catalogue information (MD_PortrayalCatalogueReference)

This package contains information identifying the Portrayal Catalogue used. It consists of the optional entity MD_PortrayalCatalogueReference. This entity contains the mandatory element used to specify which Portrayal Catalogue is used by the dataset.

8-D-1.9 Distribution information (MD_Distribution)

This package contains information about the distributor of, and options for obtaining, a resource. It contains the optional MD_Distribution entity. MD_Distribution is an aggregate of the options for the digital distribution of a dataset (MD_DigitalTransferOptions), identification of the distributor (MD_Distributor) and the format of the distribution (MD_Format), which contain mandatory and optional elements. MD_DigitalTransferOptions contains the medium used for the distribution (MD_Medium) of a dataset, and is an aggregate of MD_DigitalTransferOptions. MD_Distributor is an aggregate of the process for ordering a distribution (MD_StandardOrderProcess).

The “distributionFormat” role of MD_Distribution is mandatory if the “distributorFormat” role of MD_Distributor is not documented. The “distributorFormat” role of MD_Distributor is mandatory if the “distributionFormat” role of MD_Distribution is not documented.

8-D-1.10 Metadata extension information (MD_MetadataExtensionInformation)

This package contains information about user specified extensions. It contains the optional MD_MetadataExtensionInformation entity. MD_MetadataExtensionInformation is an aggregate of information describing the extended metadata elements (MD_ExtendedElementInformation).

8-D-1.11 Application Schema information (MD_ApplicationSchemaInformation)

This package contains information about the Application Schema used to build a dataset. It contains the optional entity MD_ApplicationSchemaInformation which is an aggregate of MD_SpatialAttributeSupplement, which is an aggregate of MD_FeatureTypeList. The entities contain mandatory and optional elements.

Metadata extensions for Imagery from ISO 19115-2. The work on ISO 19115-2 is still (June 2009) in the development phase. However the general types of extensions have been identified. The following are examples of those extensions.

MI_AcquisitionInformation – a new class in the Data Identification Package

1) planningPoints
2) instrumentIdentification
3) platformIdentification
4) missionIdentification

MD_ImageDescription

1) aerotriangulationReference
2) localElevationAngle
3) localAzimuthAngle
4) relativeAzimuth
5) platformDescending
6) nadir

Other metadata will derive from the work on ISO 19130 Sensor Models, and any input from IHO. In particular there is a need for input on metadata about hydrographic sounding sensors.

Appendix 8-E
Feature Oriented Images
(informative)

The Spatial Object in the S-100 model and in the ISO model can represent either Vector data or Imagery, Gridded or Coverage data. Both make reference to an externally defined Spatial Referencing System. Also both are feature oriented.

Most people do not think of Imagery, Gridded or Coverage data as being feature oriented. At the minimum an image or a set of gridded measurements or a TIN coverage can be considered as a single feature, so in essence such data is feature oriented. But this is the minimum case. It is possible to include in an imagery, gridded or coverage dataset a data structure that could group pixels to identify features. For example an attribute could be included with each pixel that carried a feature ID number. This would allow one to identify certain features as being a particular feature type. In an image dataset corresponding to a scanned paper chart, one could mark sets of pixels as representing various hydrographic features. There are other more efficient methods of carrying such feature ID attribute data for an image than adding bits to each pixel. There is no obligation to build such sophisticated feature oriented imagery datasets, but both S-100 and the ISO standards allow them to be built if needed. This can be very important for the fusion of bathymetric sensor data represented as an image together with vector chart data.

This appendix discusses the utility of feature oriented images and gives examples. The structures to support feature oriented images are very simple and are part of the application. It is not obvious that a single reference within the data model allows for an entire capability, so this informative appendix illustrates how that capability can be implemented and used.

All gridded datasets are feature oriented, in that a coverage is a subtype of a feature. That is, an entire gridded dataset can be considered to be a single feature. A feature structure can be applied to gridded data in two different ways. First, a discrete coverage can carry a feature code as an attribute. For example, a coverage corresponding to the postal code system will have discrete values for each postal code, yet still cover the country completely. The only difference in the Application Schema is a relationship between the discrete coverage and the feature. This is shown in Figure 8-E 1.

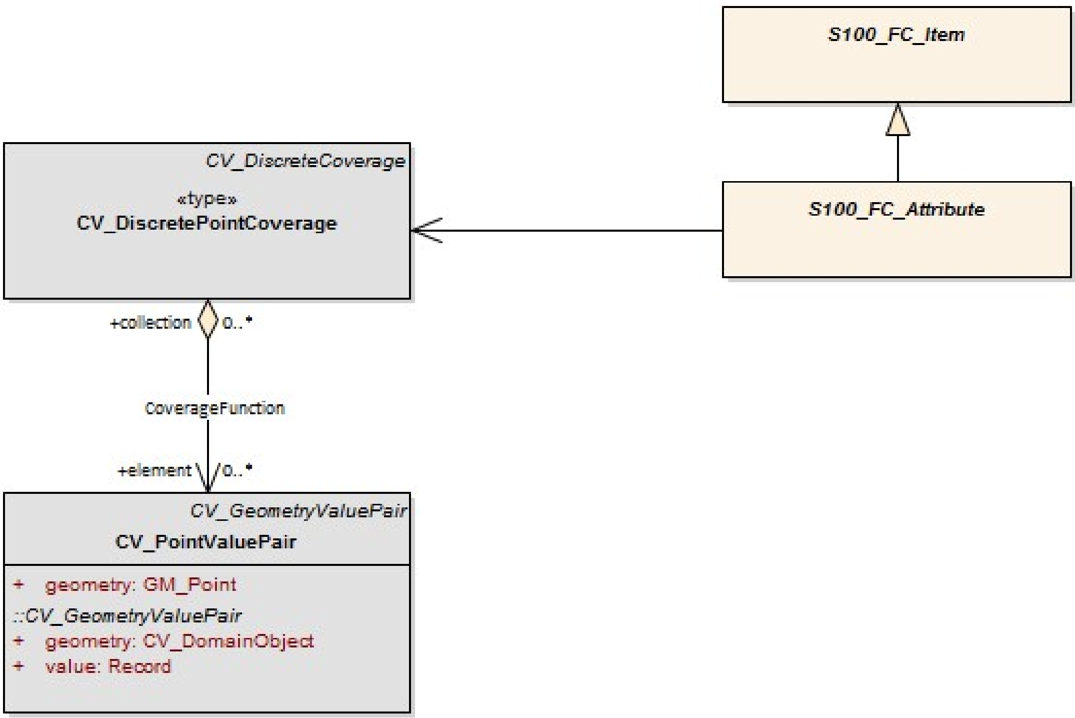


Figure 8-E-1 – Feature oriented discrete coverage

The model shown in Figure 8-E-2 below illustrates the collocation of two grids, supported by one grid value matrix to achieve the assignment of feature ID to specific cells. The discrete coverage allows for the assignment of feature codes to Grid Value Matrix entities and the continuous coverage allows one to handle the image.

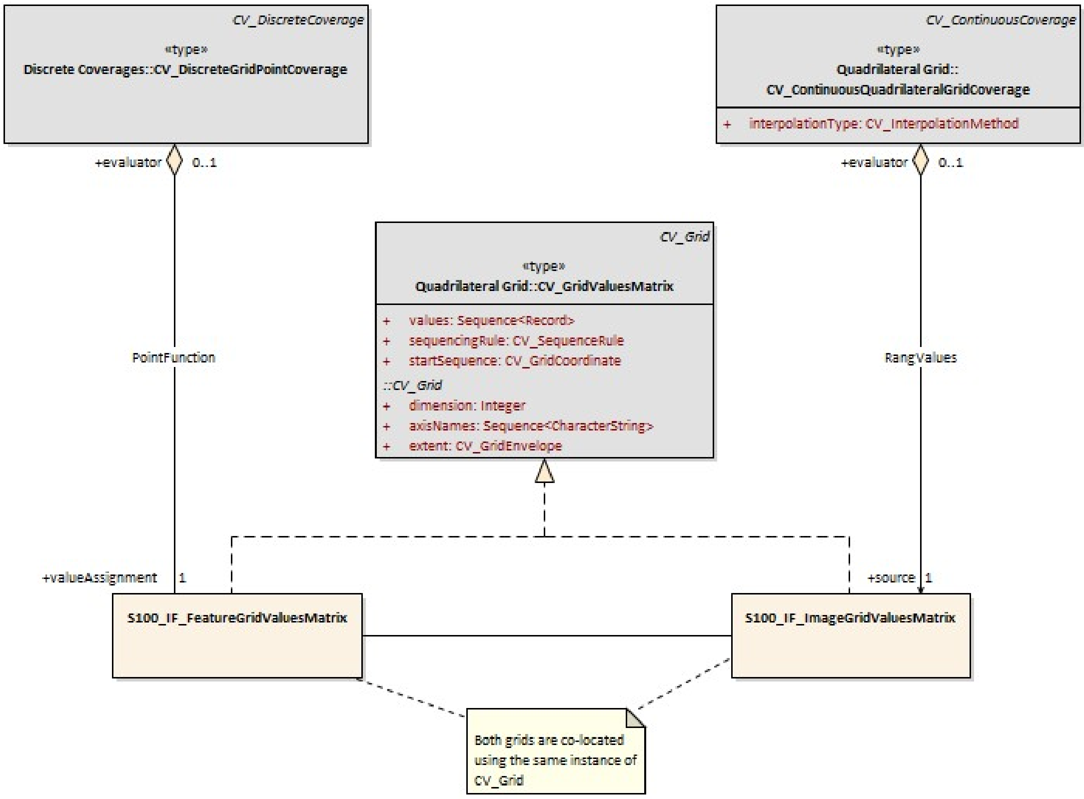


Figure 8-E-2 – Assigning feature codes to pixels in an image.

The second method of establishing a feature structure is to develop a composite dataset that contains many separate but adjoining coverages. The coverages may be continuous or discrete. This is very much like the way a "vector" dataset is composed where each feature has its own geometry and attributes. In fact vector data may be mixed with coverage data in the same dataset. The Application Schema simply allows multiple instances of features.

Geometric elements such as grids may be shared between multiple features, and features may be related by composition or other relationships as allowed in the general feature model of ISO 19109. A complex feature may include both a continuous grid coverage and vector data such as a polygonal boundary. A feature oriented dataset may contain both a continuous coverage of the ocean as collected by sonar, and point and line features corresponding to navigational aids. Topological primitives may relate all of the features. This allows for some interesting and useful structures. For example, a scanned paper map represented as a gridded dataset may include additional vector data describing the roads and other features on the scanned map, which is not "visible" in that it is not portrayed, but which is active in that a user might query the name of a feature or traverse along a road on what would appear to be a gridded data set.

S-100 – Part 9

Portrayal

Page intentionally left blank

Contents

Page intentionally left blank

9-1 Scope

This part of the standard defines the models, structures and formats for a machine readable Portrayal Catalogue. The intent is for a Portrayal Catalogue to be delivered separately from product datasets such that it can be imported and interpreted to map Feature objects defined according to the Part 3 General Feature Model (GFM) into Drawing Instructions and symbolization.

The actual contents of a Portrayal Catalogue need to be defined as part of a Product Specification using the mechanism and structures defined in this part. For example a Product Specification would include a set of mapping rules, a set of symbols, linestyles, colours etc and make it available for use with product datasets.

This part includes mechanisms for portrayal of 2D vector data according to the GFM as well as Coverage data. It does not include drawing instructions and symbol structures intended for 3D portrayal. It does not include the generation of pick reports or textual reports however the approach of exposing the content to mapping rules could be implemented to generate textual or html formatted output.

9-2 Conformance

This part of the specification conforms to ISO 19117:2012 (E) according to the Annex A Abstract test suite.

9-3 Normative references

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

ICC Specification Version 4 – International Color Consortium

ISO 19117: 2012 (E), *Geographic Information – Portrayal*

W3C.REC-XSLT-1.0-19991116, *XSL Transformations (XSLT) Version 1.0*, *W3C Recommendation* 16 November 1999, <http://www.w3.org/TR/xslt>

W3C.REC-SVGTiny12-20081222, *Scalable Vector Graphics (SVG) Tiny 1.2 Specification*, *W3C Recommendation* 22 December 2008, <http://www.w3.org/TR/2008/REC-SVGTiny12-20081222>

W3C.REC-CSS2-20110607, *Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification, W3C Recommendation* 07 June 2011, <http://www.w3.org/TR/2011/REC-CSS2-20110607>

TrueType-1.66-1995, *True Type Font Revision 1.66* 1995, <http://www.microsoft.com/typography/SpecificationsOverview.mspx>

9-4 Portrayal Catalogue

This part of the standard describes a Portrayal Catalogue and its contents. The concept in this standard is that feature data is modelled with a focus on content and portrayal of a feature is accomplished using rules or functions that map the content to the appropriate symbols and display characteristics. This concept allows the same content to be displayed in different ways and allows the display mapping rules to be maintained without having to modify all the content data.

The Portrayal Catalogue contains portrayal functions that map the features to symbology it also contains symbol definitions, colour definitions, portrayal parameters and portrayal management concepts such as viewing groups. The goal in S-100 is to provide a mechanism where, for a given product, the Portrayal Catalogue can be delivered as data in a machine readable form such that a compliant implementation can display the product feature data using the given Portrayal Catalogue.

For the purposes of this Part, the S-100 and ISO type “CharacterString” and the XML schema type “string” are to be treated as equivalent.

9-5 General portrayal model

The general portrayal model is illustrated in figure 9.1.

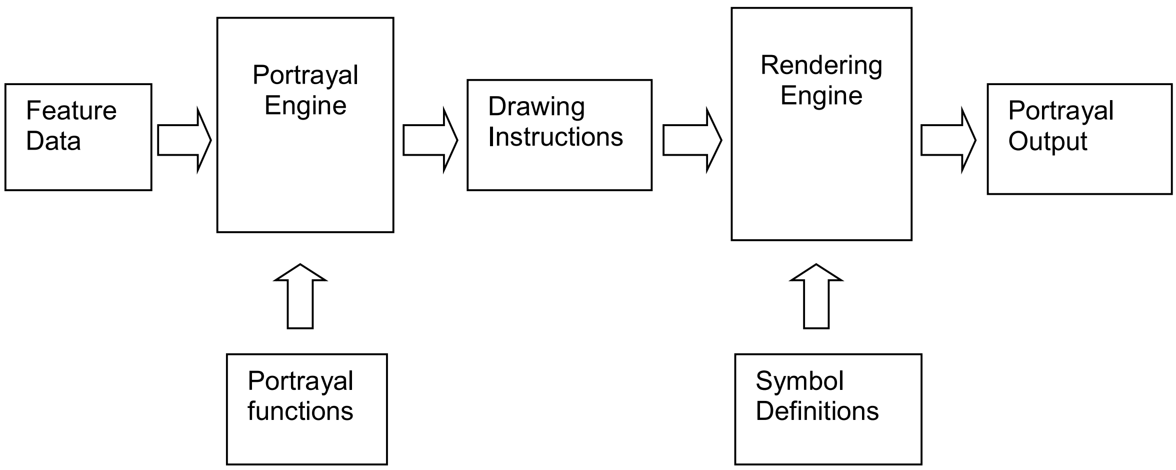


Figure 9-1 — General portrayal model

This part of S-100 defines a feature-centred function-based portrayal mechanism. Instances of features are portrayed based on portrayal functions, which make use of geometry and attribute information. The relationship between the feature instances, attributes, and the underlying spatial geometry is specified in a Product Specification based on the General Feature Model of S-100.

Portrayal information is needed to portray a dataset containing geographic data. The portrayal information is defined as drawing instructions created by specific portrayal functions. The portrayal mechanism makes it possible to portray the same dataset in different ways without altering the dataset itself.

The drawing instructions are intermediate data used by the rendering engine to produce the portayal output. During the rendering process, the rendering engine uses the symbol definitions to create the output according to the output device.

The symbol definitions contain the details of all graphical elements used for the portrayal. The model of the symbol definitions is described in this document.

9-5.1 The portrayal process

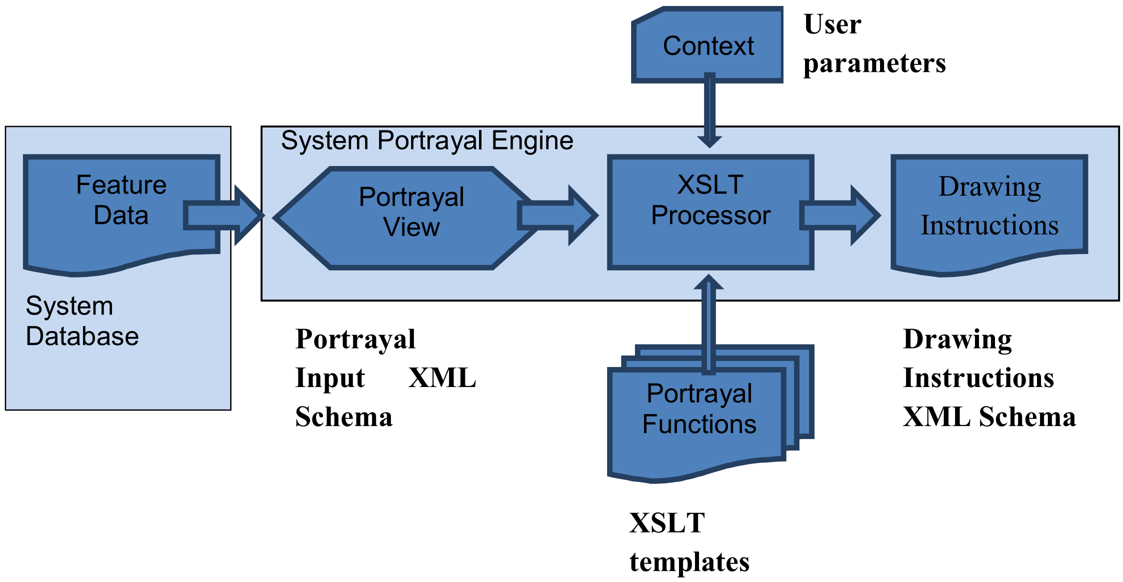


Figure 9-2 — Portrayal process

The system has Feature data within its internal database that needs to be portrayed. The System Portrayal Engine transforms the Feature data into drawing instructions. Drawing instructions include such things as references to symbol definitions, priority and filtering information. The drawing instructions are further processed by the rendering engine to produce the final display.

In this process, feature data needs to be exposed to the XSLT processor as XML content. The XSLT processor applies the best matching template or portrayal function to each feature. The portrayal function uses the defined logic to transform the input feature content along with related context information into drawing instructions which are output as XML.

The functionality of the System Portrayal Engine is defined in terms of XSLT. XSLT is a declarative language. An XSLT processor transforms XML input into XML output. Contextual and user parameters can be fed into the XSLT processor for use by the portrayal functions. Portrayal functions in XSLT can range from simple lookup or best match templates to complex conditional logic. XSLT is defined to work on an XML node tree however there are implementations that interface the XSLT processor directly with internal structures or relational database tables. Although there are newer versions of XSLT, XSLT 1.0 (http://www.w3.org/TR/xslt) has been chosen for this portrayal specification as the most commonly supported.

This portrayal specification defines how machine readable portrayal transformation functions are implemented as XSLT templates disseminated in XSL files. Since XSLT is defined to operate on XML and produce XML the XML input and output are described as part of this specification. A conformant System Portrayal Engine must operate consistently with XSLT in order to process the machine readable XSL files and produce equivalent output.

9-6 Package overview

The following diagram shows the packages for implementing this standard.

The InputSchema describes how data is presented to the portrayal engine (XSLT processor). The Presentation package includes two subpackages one describing the Portrayal Catalogue structure the other describing the drawing instructions. Drawing instructions are the output of the portrayal engine (XSLT processor).

The SymbolDefinitions package describes the graphic primitives used for portrayal.

The portrayal engine is using standard XSLT. There is no package describing this part of the portrayal.

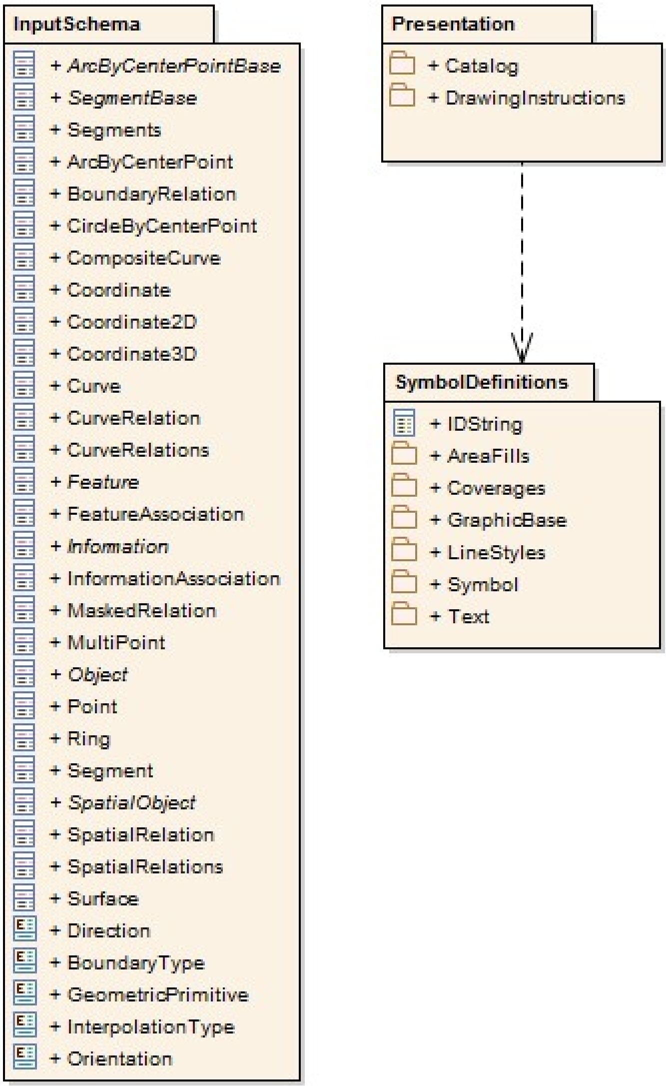


Figure 9-3 — Packages

9-7 Data input Schema

9-7.1 Introduction

The data input Schema describes how data should be presented to the XSLT processor. Encoded data can be transformed to an XML document or a presentation of such a document, for example a DOM-tree. It is also possible to model the data to look like XML and use a special software interface to present such data to the XSLT processor.

Whichever method is used, this Schema describes how the data must be organized. In this standard only the base types are described. These types should be used along with a product's Feature Catalogue to generate the input XML for portrayal processing, as described in Annex A.

NOTE It is assumed for the examples in this section that types of this Schema are in the namespace s100.

9-7.2 Enumerations

For the use in this Schema the following enumeration types are defined:

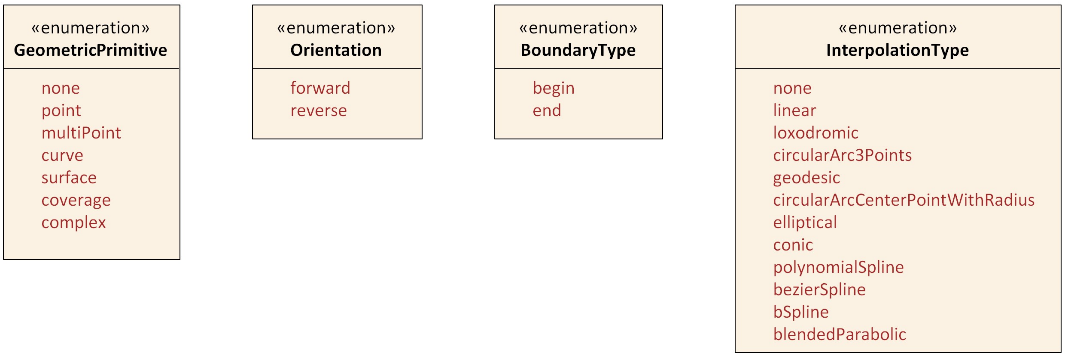


Figure 9-4 — Input Schema Enumerations

GeometricPrimitive

This enumeration describes the type of geometric primitive that is used by a feature object. If the feature object uses different geometric primitives the value `Complex` has to be used.

```
<xs:simpleType name="GeometricPrimitive">
  <xs:restriction base="xs:string">
    <xs:enumeration value="None"/>
    <xs:enumeration value="Point"/>
    <xs:enumeration value="MultiPoint"/>
    <xs:enumeration value="Curve"/>
    <xs:enumeration value="Surface"/>
    <xs:enumeration value="Coverage"/>
    <xs:enumeration value="Complex"/>
  </xs:restriction>
</xs:simpleType>
```

Orientation

The enumeration `Orientation` is used to specify the orientation of a referenced geometry that is used by a feature object or by a complex curve.

```
<xs:simpleType name="Orientation">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Forward"/>
    <xs:enumeration value="Reverse"/>
  </xs:restriction>
</xs:simpleType>
```

BoundaryType

This enumeration describes the type of a topologic boundary.

```
<xs:simpleType name="BoundaryType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Begin"/>
    <xs:enumeration value="End"/>
  </xs:restriction>
</xs:simpleType>
```

InterpolationType

This enumeration describes the mathematical interpolation method between two control points in a line segment. Note that the methods depend on the underlying coordinate reference system and not all of them are valid for all types of CRS. The Product Specification should specify the details of the use of interpolation.

```
<xs:simpleType name="InterpolationType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="None"/>
    <xs:enumeration value="Linear"/>
    <xs:enumeration value="Loxodromic"/>
    <xs:enumeration value="CircularArc3Points"/>
    <xs:enumeration value="Geodesic"/>
    <xs:enumeration value="CircularArcCenterPointWithRadius"/>
    <xs:enumeration value="Elliptical"/>
    <xs:enumeration value="Conic"/>
    <xs:enumeration value="PolynomialSpline"/>
    <xs:enumeration value="BezierSpline "/>
    <xs:enumeration value="BSpline"/>
    <xs:enumeration value="BlendedParabolic"/>
  </xs:restriction>
</xs:simpleType>
```

9-7.3 Coordinates

In case that coordinates have to be presented to the XSLT processor the following types have to be used.

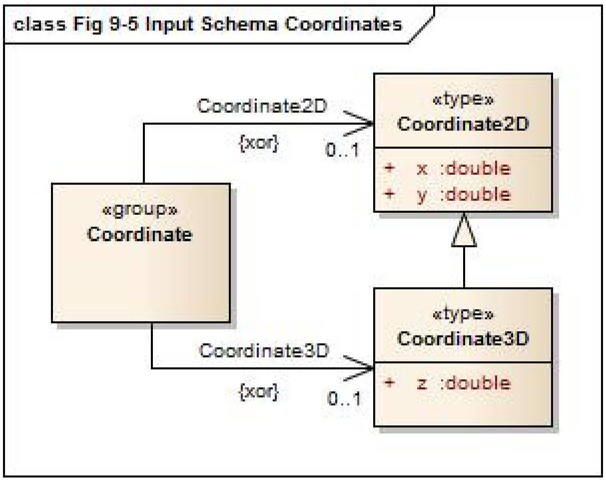


Figure 9-5 — Input Schema Coordinates

The types Coordinate2D and Coordinate3D are for a simple coordinate tuple. They are defined as:

```
<xs:complexType name="Coordinate2D">
  <xs:sequence>
    <xs:element name="x" type="xs:double"/>
    <xs:element name="y" type="xs:double"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="Coordinate3D">
  <xs:complexContent>
    <xs:extension base="Coordinate2D">
      <xs:sequence>
        <xs:element name="z" type="xs:double"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Note that the type Coordinate3D is an extension of the type Coordinate2D.

Example:

```
<s100:Coordinate2D>
  <s100:x>9.12345</s100:x>
  <s100:y>52.56789</s100:y>
</s100:Coordinate2D>
```

And

```
<s100:Coordinate2D>
  <s100:x>9.12345</s100:x>
  <s100:y>52.56789</s100:y>
  <s100:z>12.5</s100:z>
</s100:Coordinate2D>
```

A group Coordinate is defined where coordinate tuples can be used. The use of 2D or 3D tuples is mutually exclusive.

```
<xs:group name="Coordinate">
   <xs:choice>
      <xs:element name="Coordinate2D" type="Coordinate2D"/>
      <xs:element name="Coordinate3D" type="Coordinate3D"/>
   </xs:choice>
</xs:group>
```

9-7.4 Associations

According to the general feature model there are two types of associations:

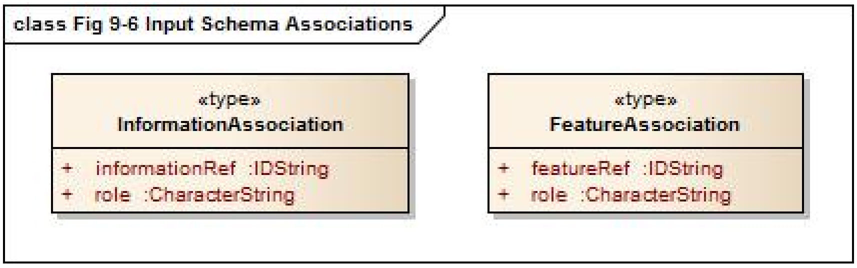


Figure 9-6 — Input Schema Associations

For each association a separate type is defined in the Schema:

```
<xs:complexType name="InformationAssociation">
   <xs:attribute name="informationRef" type="IDString" use="required"/>
   <xs:attribute name="role" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="FeatureAssociation">
   <xs:attribute name="featureRef" type=" IDString " use="required"/>
   <xs:attribute name="role" type="xs:string" use="required"/>
</xs:complexType>
```

The attributes informationRef and featureRef correspond to the attribute id of the referenced information respective feature object. See the section on objects for more details.

Each S100_FC_InformationAssociation and S100_FC_FeatureAssociation defined in a Feature Catalogue describes a subtype of InformationAssociation and FeatureAssociation respectively. These subtypes may define additional attributes of the association; these attributes should be included in the input XML as described in Annex A.

9-7.5 Spatial relations

In the general feature model different relations are modelled between feature types and spatial types but also between spatial types. For such relations the following types are defined by this Schema.

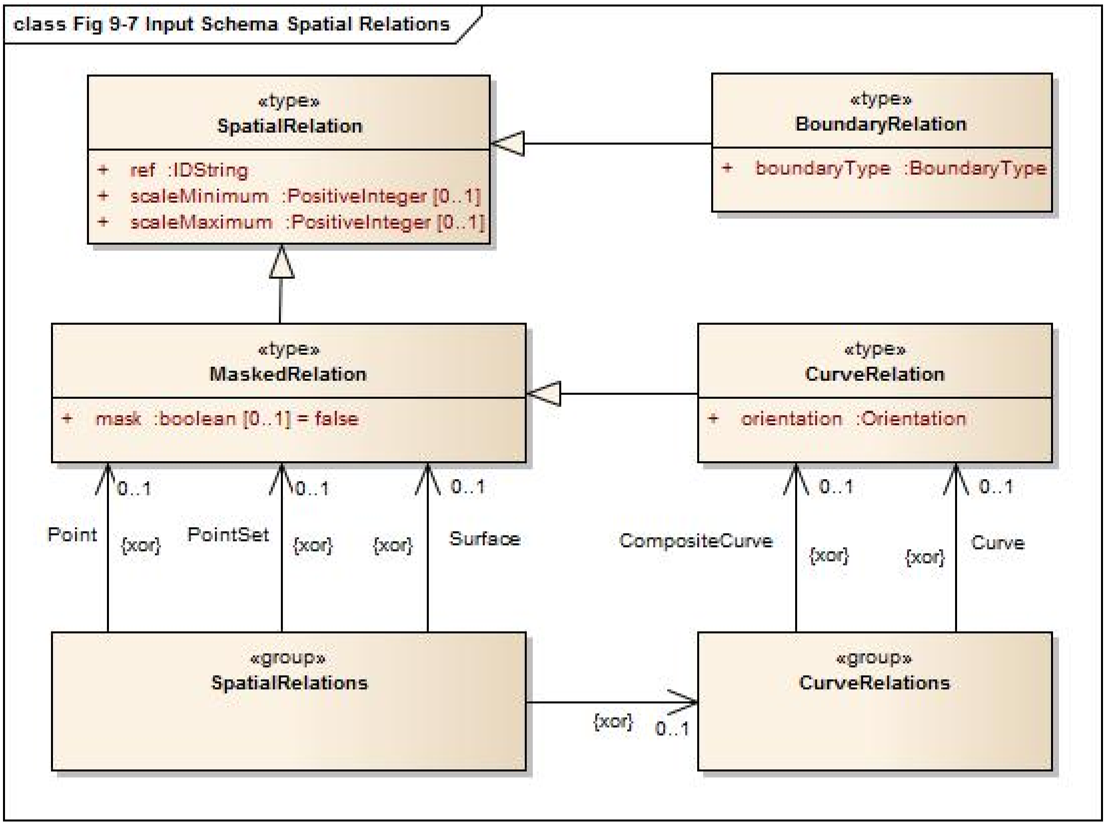


Figure 9-7 — Input Schema Spatial Relations

The type SpatialRelation is the base type for all relations to spatial objects. The ref attribute corresponds to the attribute id of the spatial object.

```
<xs:complexType name="SpatialRelation">
  <xs:attribute name="ref" type=" IDString " use="required"/>
  <xs:attribute name="scaleMinimum" type="xs:positiveInteger" use="required"/>
  <xs:attribute name=" scaleMaximum " type="xs:positiveInteger" use="required"/>
</xs:complexType>
```

The other relation types are derived from this type and add information according to the specific use of that relation. The type MaskedRelation adds an attribute mask that specifies if a referenced spatial object should not be used for portrayal.

```
<xs:complexType name="MaskedRelation">
  <xs:complexContent>
    <xs:extension base="SpatialRelation">
      <xs:attribute name="mask" type="xs:boolean" default="false"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Note that the attribute mask is not mandatory but has a default value for the case of its absence.

The type BoundaryRelation adds a boundary type to the relation and is used when the relation describes a topological relation, for example the relation to a bounding node of a curve.

```
<xs:complexType name="BoundaryRelation">
  <xs:complexContent>
    <xs:extension base="SpatialRelation">
```

```
      <xs:attribute name="boundaryType" type="BoundaryType" use="required"/>
    </xs:extension>
   </xs:complexContent>
</xs:complexType>
```

The type CurveRelation is used whenever a curve is referenced by a spatial relation since it is necessary to specify if the curve is used in the same direction as it is defined or in the reverse order. The type is derived from MaskedRelation since each curve can be a subject of masking.

```
<xs:complexType name="CurveRelation">
  <xs:complexContent>
    <xs:extension base="MaskedRelation">
      <xs:attribute name="orientation" type="Orientation" use="required"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Two groups are defined for Spatial relations. One group defines the possible relations to curves; the other defines all possible spatial relations.

```
<xs:group name="CurveRelations">
  <xs:choice>
    <xs:element name="Curve" type="CurveRelation"/>
    <xs:element name="CompositeCurve" type="CurveRelation"/>
  </xs:choice>
</xs:group>

<xs:group name="SpatialRelations">
  <xs:choice>
    <xs:element name="Point" type="MaskedRelation"/>
    <xs:element name="PointSet" type="MaskedRelation"/>
    <xs:element name="Surface" type="MaskedRelation"/>
    <xs:group ref="CurveRelations"/>
  </xs:choice>
</xs:group>
```

How these groups are used is demonstrated in Annex A.

9-7.6 Objects

All objects in a data set are based on the type Object which carries the common properties of all objects. Each object needs to be uniquely identifiable within a data set. This is done by the attribute id.

An optional association to `Information` is provided. When an information association is present, the code of an information association defined in the Feature Catalogue should be used as the element name in place of *associatedInformation* - see Appendix 9-A for more information

```
<xs:complexType name="Object" abstract="true">
  <xs:sequence>
      <xs:element name="associatedInformation" type="InformationAssociation" minOccurs="0"
        maxOccurs="unbounded"/>
      <!—When generating input XML the base schema should be extended to provide a substitution group
        for each information association defined in the Feature Catalogue. For example,  if the Feature
        Catalogue defines an information association having a code of SpatialAssociation, the base schema
        should be extended with:
         <xs:element name="SpatialAssociation" substitutionGroup="associatedInformation"/>
     -->
  </xs:sequence>    <xs:attribute name="id" type="IDString" use="required"/>
</xs:complexType>
```

Note that the type of the identifier is IDString to be as general as possible with respect to different methods used for identification. The characters allowed in this string are 0-9a-zA-Z.

```
<xs:simpleType name="IDString">
        <xs:restriction base="xs:string">
                <xs:minLength value="1"/>
                <xs:pattern value="[0-9a-zA-Z_]*"/>
        </xs:restriction>
</xs:simpleType>
```

The model of all objects is given in Figure 9-8 below.

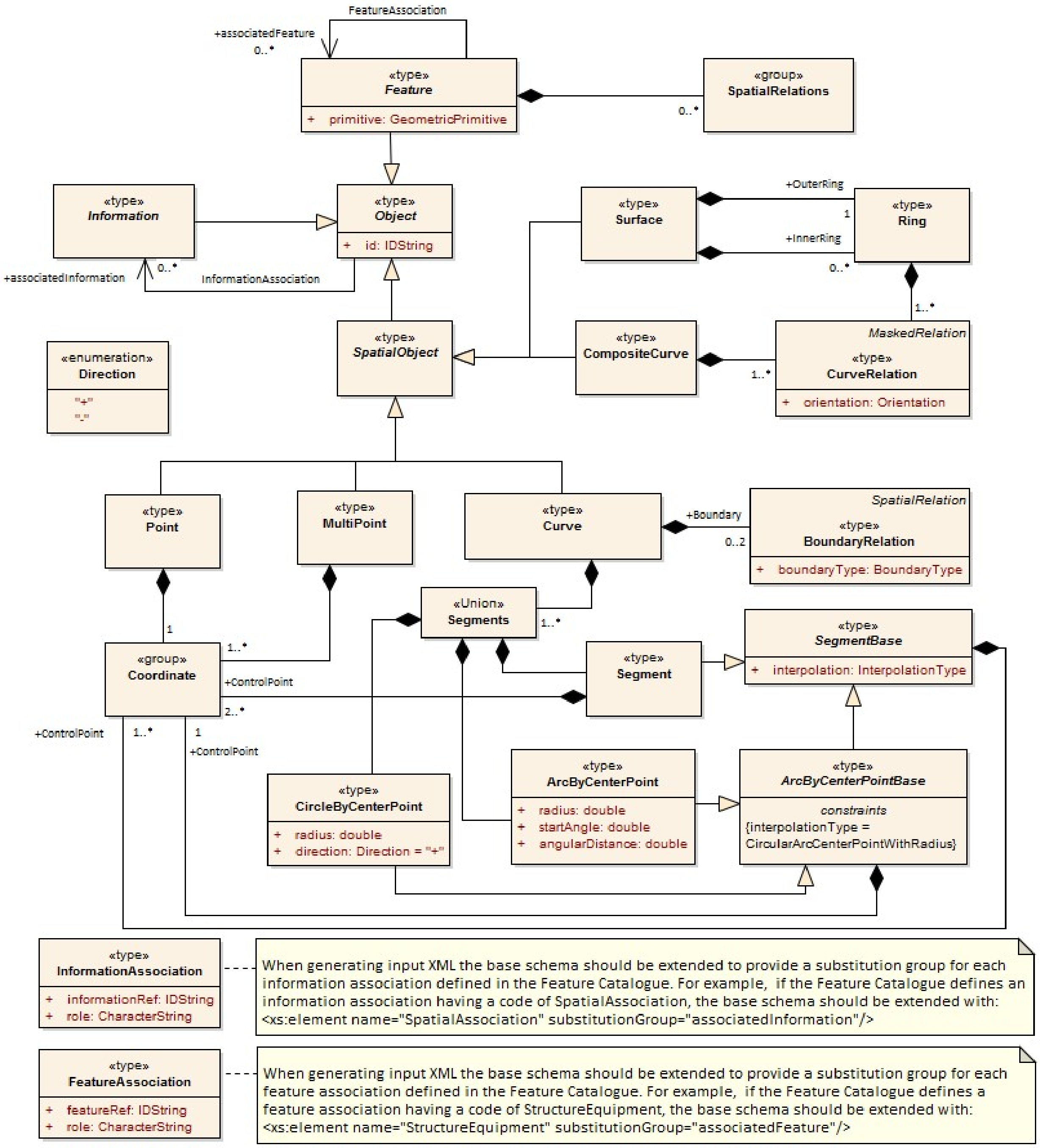


Figure 9-8 — Input Schema Objects

9-7.7 Spatial objects

9-7.7.1 Preface

Spatial objects in a data set carry the geometric location of a feature object. The following types are supported by this standard:

- Point
- MultiPoint
- Curve
- Composite curve
- Surface

Attributes are not permitted for spatial objects by the GFM. All types are derived from the type Object, meaning they have an identifier and may have associated information such as spatial quality.

9-7.7.2 SpatialObject

SpatialObject is an abstract class which serves as the base class for all spatial objects.

```
<xs:complexType name="SpatialObject" abstract="true">
  <xs:complexContent>
    <xs:extension base="Object">
  </xs:complexContent>
</xs:complexType>
```

9-7.7.3 Point

A point carries a single coordinate tuple, 2D or 3D. The definition looks like.

```
<xs:complexType name="Point">
  <xs:complexContent>
    <xs:extension base="SpatialObject">
      <xs:sequence>
        <xs:group ref="Coordinate"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Note that the group Coordinate is used within the definition to allow both Coordinate2D and Coordinate3D elements

9-7.7.4 MultiPoint

Similar to Point this type defines point geometry for a feature object. The difference is that a set of tuples can be defined. Therefore maxOccurs is set to unbounded.

```
<xs:complexType name="MultiPoint">
  <xs:complexContent>
    <xs:extension base="SpatialObject">
      <xs:sequence>
        <xs:group ref="Coordinate" minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

9-7.7.5 Curve

Curves describe the line geometry of a feature object. They are made of segments where each segment has a sequence of control points and an interpolation method. The latter defines the geometry between the control points according to the used coordinate reference system. There are two special types of segments:

1. ArcByCenterPoint
   A circular arc defined by a center point and a radius. The beginning of the arc is defined by the start angle and the length of the arc is defined by the angular length. This length is a signed quantity defining the direction of the arc: positive means clockwise.

2. CircleByCenterPoint
   A circle defined by a center point and a radius.

The abstract type SegmentBase defines a sequence of control points and the attribute for the interpolation type:

```
<xs:complexType name="SegmentBase" abstract="true">
  <xs:sequence>
    <xs:element name="ControlPoint" type="Coordinate2D" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="interpolation" type="InterpolationType" use="required"/>
</xs:complexType>
```

From this type the type Segment is derived by restricting the number of control points to at least two:

```
<xs:complexType name="Segment">
  <xs:complexContent>
    <xs:restriction base="SegmentBase">
      <xs:sequence>
        <xs:element name="ControlPoint" type="Coordinate2D" minOccurs="2" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:restriction>
  </xs:complexContent>
</xs:complexType>
```

For the “by center point” segments the abstract base type is also derived from SegmentBase restricting the number of control points to exact 1 and fixes the value of the attribute interpolation.

```
<xs:complexType name="ArcByCenterPointBase" abstract="true">
  <xs:complexContent>
    <xs:restriction base="SegmentBase">
      <xs:sequence>
        <xs:element name="ControlPoint" type="Coordinate2D" minOccurs="1" maxOccurs="1"/>
      </xs:sequence>
      <xs:attribute name="interpolation" type="InterpolationType" use="required"
                    fixed="CircularArcCenterPointWithRadius"/>
    </xs:restriction>
  </xs:complexContent>
</xs:complexType>
```

The ArcByCenterPoint is then an extension of this type adding attributes for radius, start angle and angular length.

```
<xs:complexType name="ArcByCenterPoint">
  <xs:complexContent>
    <xs:extension base="ArcByCenterPointBase">
      <xs:attribute name="radius" type="xs:double" use="required"/>
      <xs:attribute name="startAngle" type="xs:double" use="required"/>
      <xs:attribute name="angularDistance" type="xs:double" use="required"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The CircleByCenterPoint type is very similar. The attribute start angle is not defined since it is meaningless. The direction is here defined by the attribute direction which has values ‘+’ or ‘-‘.

```
<xs:simpleType name="Direction">
  <xs:restriction base="xs:string">
    <xs:enumeration value="+"/>
    <xs:enumeration value="-"/>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="CircleByCenterPoint">
  <xs:complexContent>
    <xs:extension base="ArcByCenterPointBase">
      <xs:attribute name="radius" type="xs:double" use="required"/>
      <xs:attribute name="direction" type="Direction" default="+"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

A group is defined that allows the use of the different type of segments.

```
  <xs:group name="Segments">
    <xs:choice>
      <xs:element name="Segment" type="Segment"/>
      <xs:element name="ArcByCenterPoint" type="ArcByCenterPoint"/>
      <xs:element name="CircleByCenterPoint" type="CircleByCenterPoint"/>
    </xs:choice>
  </xs:group>
```

The type Curve finally combines a sequence of segments with the topological boundary. The topological boundary of a curve is the beginning and end node implemented by a Point object.

```
<xs:complexType name="Curve">
  <xs:complexContent>
    <xs:extension base="SpatialObject">
      <xs:sequence>
        <xs:element name="Boundary" type="BoundaryRelation" minOccurs="0" maxOccurs="2"/>
        <xs:group ref="Segments" minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

9-7.7.6 CompositeCurve

A composite curve describes the line geometry of a feature object just like a ‘simple’ curve. But instead of using coordinates to define the geometry it is using a sequence of other curves, including other composite curves. With other words it is a sequence of relations to other curves.

```
<xs:complexType name="CompositeCurve">
  <xs:complexContent>
    <xs:extension base="SpatialObject">
      <xs:sequence>
        <xs:group ref="CurveRelations" minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

9-7.7.7 Surface

Surfaces describe the area geometry of a feature object. The surface itself is defined by its boundary. The boundary consists of an outer ring and optionally a number of inner rings. The inner rings describe holes in the area. Each ring is a closed polygon made from one or many curves. That means that a ring is very similar to a composite curve but unlike the composite curve it is not derived from `SpatialObject` because it does not need to be identifiable. The definition of a ring simply looks like:

```
<xs:complexType name="Ring">
  <xs:group ref="CurveRelations" minOccurs="1" maxOccurs="unbounded"/>
</xs:complexType>
```

And the definition of a surface finally is:

```
<xs:complexType name="Surface">
  <xs:complexContent>
    <xs:extension base="SpatialObject">
      <xs:sequence>
        <xs:element name="OuterRing" type="Ring"/>
        <xs:element name="InnerRing" type="Ring" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

9-8 Information objects

Information object are identifiable and sharable pieces of information within a data set. In the model an abstract type `Information` is derived from the type `Object`. Although no additional properties are added, this type is useful for semantic reasons.

```
<xs:complexType name="Information" abstract="true">
  <xs:complexContent>
    <xs:extension base="Object"/>
  </xs:complexContent>
</xs:complexType>
```

9-9 Feature objects

Feature objects are abstractions of real world phenomena. This Schema defines the abstract base type for any feature type. The type Feature is derived from Object and adds a sequence of spatial relations and a geometric primitive attribute to the properties from the base class, as well as optional associations to other features.

When an association is present, the code of an feature association defined in the Feature Catalogue should be used as the element name in place of *associatedFeature* - see Annex A for more information.

```
<xs:complexType name="Feature" abstract="true">
  <xs:complexContent>
    <xs:extension base="Object">
      <xs:sequence>
        <xs:group ref="SpatialRelations" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="associatedFeature" type="FeatureAssociation" minOccurs="0"
         maxOccurs="unbounded"/>
        <!—When generating input XML the base schema should be extended to provide a substitution group
         for each feature association defined in the Feature Catalogue. For example,  if the Feature Catalogue
         defines a feature association having a code of StructureEquipment, the base schema should be
         extended with:
          <xs:element name="StructureEquipment" substitutionGroup="associatedFeature"/>
        -->
      </xs:sequence>
      <xs:attribute name="primitive" type="GeometricPrimitive" use="required"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

9-10 Portrayal processing

This specification is referencing XSLT 1.0 which is a W3C recommendation, http://www.w3.org/TR/xslt

XSLT uses XPath 1.0 to address components of a document. http://www.w3.org/TR/xpath/

XSLT (XSL Transformations) is a language expressed as a well formed XML document. The intended purpose of using XSLT in portrayal is to transform the data into drawing instructions. Since XSLT is expressed in XML it is useful for interchange as a machine readable transformation language. XSLT is widely used across many domains but is perhaps most commonly used to transform XML documents into HTML for web page displays. There are many tutorials, books and reference material available for XSLT. There are also several web sites where questions can be posted and examples can be found.

XSLT uses templates to process nodes in the input XML tree and generate nodes as output XML, other SGML formats or even plain text. There are two types of templates a matching template and a named template.

Matching templates use a matching expression using XPATH to specify what elements in the input document should be processed by that template. XPATH (XML Path Language) is an expression language used to address or find components in an XML document. The path capability makes it especially useful when dealing with a hierarchy of content such as nested complex attributes. Only one matching template can match an element from the input document. Matching templates have a built in priority calculation and conflict resolution method that is used to determine which one to use in the case where multiple templates match the same element. Priority numbers can be explicitly assigned as an attribute of a matching template in order to override the default conflict resolution behaviour.

Named templates are called by another template along with the data to be processed. Named templates can also have parameters. These are useful for formatting or other operations that are commonly used in a transformation. A named template can even call itself (recursion), which can be useful for operations such as string token parsing.

A template can loop over a set of nodes that match an XPath expression using an “xsl:apply-templates “ or “xsl:for-each” instruction element. The nodes can also be sorted before being processed. Conditional processing is available by using a simple “xsl:if” instruction or an “xsl:choose” instruction. The choose instruction allows a set of expressions to be tested such that only the first one matching is processed and if no match is found an optional otherwise statement is used to handle a default. This is useful for testing enumerated data such that a different output is generated depending upon the enumeration value.

XSLT also includes the ability to have parameters passed at the top level and accessed within any of the templates. These parameters can be useful to provide contextual information to the transformation. There are also variables in XSLT but they can only be assigned data as part of their definition, unlike other languages where variables can be reassigned. Variables are useful to collect data or decision results that can be passed as parameters to another template or used in conditional statements.

XSLT can include or import other XSLT documents. This capability can be useful for management of templates and reuse of templates by multiple top level XLST documents.

Examples

Given the example XML below

```xml
<BeaconCardinal id="2">
  <s100:Point ref="3"/>
  <categoryOfCardinalMark>3</categoryOfCardinalMark>
</BeaconCardinal>
<BeaconCardinal id="3">
  <s100:Point ref="3"/>
  <categoryOfCardinalMark>2</categoryOfCardinalMark>
```

```
</BeaconCardinal>
```

A simple matching XSLT template used as a portrayal function

```
<xsl:template match="BeaconCardinal">
        <!—This is a comment. This template matches a BeaconCardinal node and the body of the template can
          examine data and output results -->

</xsl:template>
```

The above template will be used to process all of the BeaconCardinal objects.

The choose instruction can be used to do conditional processing within the template.

```
<xsl:template match="BeaconCardinal">

        <xsl:choose>
        <xsl:when test="categoryOfCardinalMark = '2'">
                <!-- Output symbol for BeaconCardinal categoryOfCardinalMark =2 here -->
        </xsl:when>
        <xsl:when test="categoryOfCardinalMark = '3'">
                <!-- Output symbol for BeaconCardinal categoryOfCardinalMark =3 here -->
        </xsl:when>
        <xsl:otherwise>
                <!-- Output default symbol here -->
        </xsl:otherwise>
 </xsl:choose>

</xsl:template>
```

A more advanced XPath expression can be used to refine the match.

```
<xsl:template match="BeaconCardinal[categoryOfCardinalMark=2] ">
        <!—This is a comment.  Output symbol for BeaconCardinal categoryOfCardinalMark =2 here -->

</xsl:template>
```

9-10.1 Portrayal input validation

The Portrayal Catalogue describes each valid portrayal input parameter (context parameter) and may provide associated validation rules and value constraints. The validation rules are XPath 1.0 Boolean expressions; or W3C XML Standard – Part 2, Appendix F regular expressions. The rules support validation of the user input to portrayal, and the constraints support eliminating free-form input while supporting a machine-readable user interface (UI).

To ensure user-entered values are reasonable prior to use as portrayal input, the validation rules may:

- Ensure input values are within the expected value domain;
- Ensure input values conform to an expected pattern (for example, ###-####); and
- Ensure input values are logically consistent with respect to one another.

Additionally, context parameters and validation rules can be enabled or disabled based on XPath 1.0 Boolean expressions, supporting conditional validation and a machine-readable UI. For instance, validation of the S-101 Shallow Contour parameter may be disabled when the value of the Two Shades parameter is true.

Context parameters with an associated constraint are restricted to a set of enumerated values. This allows applications to eliminate free-form input for these parameters; and enhances the machine-readable UI by associating a label with each enumerated value (for example, “On“ associated with the value “1“).

Applications support portrayal input validation by:

1. Using the constraints to restrict user input.

2. On each change to a context parameter value:
   a. Generate a simple XML document encoding the context parameter values:
      <TwoShades>true</TwoShades> <!-- etc.-->
   b. For each enabled validation of each enabled context parameter:
      i. Ensure XPath rules evaluate to true;
      ii. Ensure regular expressions match.

XPath validations should all evaluate to true, and regex validations should all match prior to processing the portrayal rules. If an enabled validation evaluates to false or doesn't match then no further portrayal processing should be performed and error messages associated with failed validation rules should be activated.

9-11 Drawing Instructions

9-11.1 The concepts of drawing instructions

9-11.1.1 General concept

The output of the portrayal engine is a set of drawing instructions, which link the feature type to a symbol and/or alert reference. The geometry is either taken from the feature type or can be generated by the portrayal functions. The latter is supported by the concept of augmented geometry.

9-11.1.2 Portrayal Coordinate Reference Systems (CRS)

In this context coordinate reference systems refer to the portrayal geometry only.

There are different CRS related to the portrayal:

- Geographic CRS
- Portrayal CRS
- Local CRS
- Line CRS
- Area CRS
- Tile CRS
- Hatch CRS

Geographic CRS are used in the geographic dataset that are portrayed. They will be mapped by means of projections and affine transformation to the Portrayal CRS. Nevertheless rotations of symbols may be still defined relative to the North-Axis of the Geographic CRS.

The Portrayal CRS defines the coordinates at the output device, for example a screen or pixmap.

Line symbols have two kinds of coordinate reference systems. The line coordinate reference system is a non-Cartesian 2-axis coordinate system. The x-axis is following the line geometry and the y-axis is perpendicular to the geometry of the curve. This CRS allows for the specification of line widths,offsets and symbols following the geometry. The second kind of coordinate reference system is a local Cartesian coordinate reference system which is defined for every location along a curve. This coordinate reference system has an x-axis that is tangential to the curve and a y-axis perpendicular to the x-axis.

An area symbol defines coordinate reference systems for its boundary and for its interior. The boundary coordinate reference systems are those defined for line symbols. The interior of the area symbol has its own coordinate reference system.

For tiled pattern and hatch patterns own CRS are defined.

9-11.1.3 Viewing Groups, Viewing Group Layers and Display Mode

The viewing group is a concept to control the content of the display. It works as an on/off switch for any drawing instruction assigned to the corresponding viewing group. The concept can be seen as a filter on the list of drawing instructions.

A drawing instruction which has multiple Viewing Groups is disabled when any assigned Viewing Group is disabled.

Viewing groups can be aggregated into Viewing Group Layers and Viewing Group Layers can be aggregated into Display Modes. Both aggregations are part of the Portrayal Catalogue.

9-11.1.4 Transparency

Additive effects of applying transparencies shall be accomplished by multiplying the component alpha values. For example, a colour token which has transparency of 10% which is drawn with a transparency of 20% shall result in (1 – 10%) * (1 – 20%) = 72% alpha = 28% transparency.

9-11.1.5 Display Planes

Display planes are a concept to split the output of the portrayal functions into separate lists. An example of this is the separation of chart information drawn under a radar image and chart information drawn over a radar image.

9-11.1.6 Display Priorities

Display priorities control the order in which the output of the portrayal functions is processed by the rendering engine. Priorities with smaller numerical values will be processed first. Instructions which have equal display priority must be ordered so that area instructions are rendered first, followed by line instructions, then point instructions, and lastly text instructions. If the display priority is equal among the same type of instruction (area, line, point, or text) some other neutral criterion must be used to order the instructions.

9-11.1.7 Null Instruction

This is an instruction to indicate that a feature is intentionally not portrayed.

9-11.1.8 Point Instruction

Overview

The Point Instruction defines the drawing of a symbol. The symbol can be parameterized. This includes rotation, scaling and offset. The details are described in the documentation of the Symbol package.

Point Geometry

When the Point Instruction references point geometry the symbol is drawn using its position.

MultiPoint Geometry

The symbol is repeated using each position of the Multi Point.

Curve Geometry

The symbol is drawn on each referenced curved from either the spatialReference or if this is not used on each curve directly referenced by the feature type. The placement of the symbol is controlled by the linePlacement element of the symbol. The details are described in the documentation of the Symbol package.

Surface Geometry

The symbol is drawn at a representative position within the surface. How this position is obtained is controlled by the areaPlacement member of the symbol. The details are described in the documentation of the Symbol package.

9-11.1.9 Line Instruction

Overview

The Line Instruction defines the drawing of a line style. Line styles include Simple and Complex Line Styles. The line style can be parameterized. The details are described in the documentation of the LineStyles package. The geometry is defined by the referenced spatial types. Only curve or surface geometry is supported. For the latter the boundary of the surface defines the geometry. The geometry defines the direction of drawing for the line style.

Suppression

When features shares curve geometry multiple line instructions may reference the same curve.

If suppression is set to true (the default) another line instruction with a higher display priority will suppress the drawing of this line instruction. If suppression is set to false this instruction cannot be suppressed.

9-11.1.10 Area Instruction

Overview

The Area Instruction defines the drawing of an area fill. Area Fills include Colour Fills and different Pattern Fills. The area fill can be parameterized. The details are described in the documentation of the AreaFills package. Only surface geometry is supported.

The area fill must include the boundary of the surface.

9-11.1.11 Text Instruction

Overview

The Text Instruction defines the drawing of text. The text can be parameterized. This includes fonts, colour and size. The details are described in the documentation of the Text package.

Point Geometry

When the Text Instruction references a point geometry the text is drawn using its position. Only TextPoint elements are supported.

MultiPoint Geometry

The text is repeated using each position of the Multi Point. Only TextPoint elements are supported.

Curve Geometry

The text is drawn on each referenced curved from either the spatialReference or if this is not used on each curve directly referenced by the feature type. Both TextPoint and TextLine elements are supported. The first is to draw text at a position on the referenced curve relative to the local CRS at that position. The latter is to draw text that follows the shape of the referenced curve. More details can be found in the documentation of the text package.

Surface Geometry

The text is drawn at a representative position within the surface. Only TextPoint elements are supported. How this position is obtained is controlled by the areaPlacement member of the TextPoint. The details are described in the documentation of the Text package.

9-11.1.12 Coverage Instruction

Overview

An instruction to portray data coverages like gridded bathymetry, satellite images, etc.

"A coverage is a feature that has multiple values for each attribute type, where each direct position within the geometric representation of the feature has a single value for each attribute type." [ISO 19123:2005, Introduction]

In this document coverage attributes used for portrayal are expected to have numeric values.

The assignment of Portrayal for a Coverage starts with a Coverage Feature. Like other Feature types a rule is used to match the Feature to Drawing instructions.

A first match lookup table is used to assign portrayal based on a specified coverage attribute. There are three options for coverage portrayal: filling with colour; annotating with numeric text; or annotating with symbols.

Discrete Coverages

Discrete coverages are portrayed by applying a symbol and/or numeric annotation to the direct position associated with each value of an attribute of the coverage.

Continuous Coverages

Continuous coverages are portrayed by filling the cells that have actual data associated, as opposed to no data (termed "fill values" in HDF5, not to be confused with colour or symbol fills as the terms are used in portrayal). The fills used in portrayal may be solid fills; patterns of symbols; pixmaps; or gradients. Fill transparency may also be specified by the applicable portrayal rule. Interpolation methods, if defined in the coverage type (see S-100 Part 8, clause 8-7.7.3) may be applied to depict variations in data values in each grid cell. The anchor point for text or symbol placement is dependent on the coverage's spatial type and the placement attribute in the Portrayal Catalogue.

Irregular shape grids, ungeorectified grids, variable cell size grids (see S-100 Part 10, Table 10c-15) are all treated similarly to regular grids as far as portrayal is concerned. For variable cell size grids, unit cells must be used for symbol fills (that is, in an expanded cell that covers more than one unit cell, the symbol must be depicted at the centre of each unit cell included in the expanded cell).

Colour Assignment

Colours are applied to a coverage by using a lookup table that matches a selected attribute value and specifies a colour. For a continuous coverage such as grid cells, pixels or tiles then each element is processed and colour filled with the appropriate colour. For a discrete coverage with distinct points colour is applied as a Pen Down or dott operation using the assigned pen width.

A lookup table entry can match a range of values and assign a single colour to that range or specify a start and end colour that is used to create a gradient or ramp effect as a linear interpolation of the value range across the colour range.

Numeric and Symbol Annotations

For a continuous coverage the centre of each cell (for example rectangle, tile, triangle) is used as the anchor point of the text or symbol.

For discrete coverages, the anchor point for annotations is the direct position associated with each value of the attribute designated by the attributeCode parameter of the CoverageFill (see clause 9-12.7.4.1).

For numeric annotations, overplot removal or collision avoidance is expected. A buffer can be used to provide some space between the annotations. A buffer of 0 means that direct overplot is used when digits interact. An enumeration called 'champion' is used to specify which annotation to keep (largest or smallest value) when an interaction occurs. For numeric annotations the text shall be placed such that the optical/geometric centre of the text represents the location.

For symbol annotations separate attributes from the coverage can be used to apply a scaling and rotation to the symbol. This can be useful for example when portraying a coverage that carries wave height and direction.

9-11.1.13 Augmented Geometry

Overview

In case the required geometry for a drawing instruction is not explicitly given in a geographic dataset the portrayal function will generate this "augmented" geometry. A set of classes for such augmented geometries are part of the model. All positions used in this classes refer to a given coordinate reference system. Three types of CRS are supported:

1. Geographic CRS
   The coordinates are geographic coordinates.
2. Portrayal CRS
   The coordinate are referenced to the output device of the portrayal.
3. Local CRS
   The coordinates referring to a coordinate system with axes parallel to the Portrayal CRS but the origin shifted to the position of the referenced feature. Only point feature are supported for that type of CRS.

Note: The generated geometry only exists temporary for the purpose of portrayal and is not part of the dataset.

All types of augmented geometries can be used for the portrayal of text.

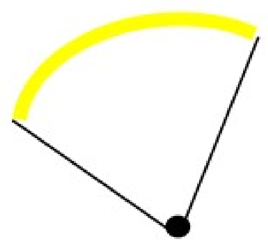


Figure 9-9 — Point Feature with Augmented geometries

More details can be found in the documentation of the drawing instruction model.

9-11.2 Model of the Drawing Instruction Package

This package contains classes which describe the output of the portrayal functions. Display instructions link the feature types and their geometry to elements from the Symbol Elements package. Figure 9-10 below shows the model.

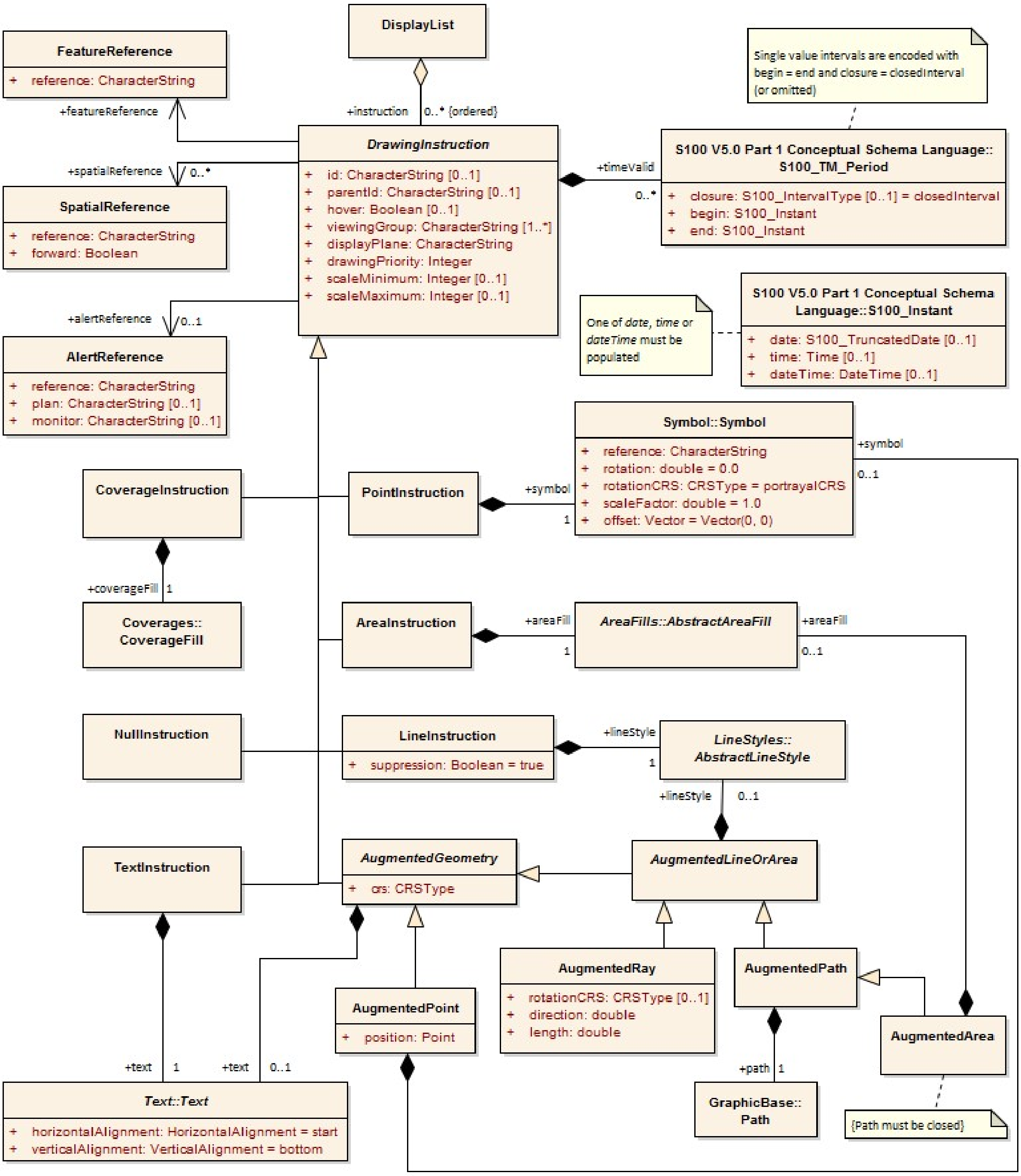


Figure 9-10 — Drawing Instructions

9-11.2.1 DisplayList

Role Name	Name	Description	Mult.	Type
Class	DisplayList	A ordered set of Drawing Instructions	-	-
Role	instruction	An instruction of this list	0..*	DrawingInstruction

9-11.2.2 DrawingInstruction

Role Name	Name	Description	Mult.	Type
Class	DrawingInstruction	Abstract base class for all drawing instructions	-	-

Role Name	Name	Description	Mult.	Type
Attribute	id	An identifier for the drawing instruction	0..1	string
Attribute	parentId	Instruction is dependent on parent drawing instruction(s). If no referenced instructions are executed during rendering then this instruction should not be executed. Execution of referenced (parent) instructions can be affected by many aspects of the visualization process including viewing group settings, display plane visibility, line suppression, scale minimum/maximum, date dependency, hover status, and dependencies of the parent instruction	0..1	string
Attribute	hover	Specifies whether the instruction is shown only on hover-over. OEM support for this feature is optional	0..1	boolean
Attribute	viewingGroup	The Viewing Group(s) the instruction is assigned to. Instruction is disabled if any Viewing Group is disabled	1..*	string
Attribute	displayPlane	The display plane the instruction is assigned to	1	string
Attribute	drawingPriority	The priority that defines the order of drawing	1	integer
Attribute	scaleMinimum	Scale denominator to define the minimum scale for which the instruction will be shown. If not given there is no minimum scale	0..1	integer
Attribute	scaleMaximum	Scale denominator to define the maximum scale for which the instruction will be shown. If not given there is no maximum scale	0..1	integer
Role	featureReference	The reference to the feature type that will be depicted by the instruction	1	FeatureReference
Role	spatialReference	The reference(s) to the spatial type components of the feature that defines the geometry used for the depiction. Not used when the entire geometry of the feature should be depicted	0..*	SpatialReference
Role	timeValid	The drawing instruction is valid during the specified time interval(s)	0..*	S100_TM_Period
Role	alertReference	The reference to the alert in the Alert Catalogue that is triggered by the geometry of the instruction	0..1	AlertReference

9-11.2.3 FeatureReference

Role Name	Name	Description	Mult.	Type
Class	FeatureReference	A reference to a feature type	-	-
Attribute	reference	The identifier of the feature type	1	string

9-11.2.4 SpatialReference

Role Name	Name	Description	Mult.	Type
Class	SpatialReference	A reference to a spatial type	-	-
Attribute	reference	The identifier of the spatial type	1	string
Attribute	forward	If true the spatial object is used in the direction in which it is stored in the data. Only applies to curves	1	boolean

9-11.2.5 AlertReference

Role Name	Name	Description	Mult.	Type
Class	AlertReference	A reference to an alert in the Alert Catalogue	-	-

Attribute	reference	The identifier of the Alert Catalogue entry	1	string
Attribute	plan	The viewing group the alert highlight is assigned to when active in route planning	0..1	string
Attribute	monitor	The viewing group the alert highlight is assigned to when active in route monitoring	0..1	string

9-11.2.6 NullInstruction

Role Name	Name	Description	Mult.	Type
Class	NullInstruction	An instruction that indicates that no portrayal is required for the referenced feature	-	-

9-11.2.7 PointInstruction

Role Name	Name	Description	Mult.	Type
Class	PointInstruction	A drawing instruction for point symbol	-	-
Role	symbol	The symbol to be depicted	1	Symbol::Symbol

9-11.2.8 LineInstruction

Role Name	Name	Description	Mult.	Type
Class	LineInstruction	A drawing instruction for line geometry	-	-
Attribute	suppression	Whether another line instruction of higher priority can suppress the drawing of this line instruction	1	boolean = *True*
Role	lineStyle	The line style used for the depiction	1	LineStyles::AbstractLineStyle

9-11.2.9 AreaInstruction

Role Name	Name	Description	Mult.	Type
Class	AreaInstruction	A drawing instruction for area geometry	-	-
Role	areaFill	The area fill used for the depiction	1	AreaFills::AbstractAreaFill

9-11.2.10 TextInstruction

Role Name	Name	Description	Mult.	Type
Class	TextInstruction	A drawing instruction for depicting text	-	-
Role	text	The text to be depicted	1	Text::Text

9-11.2.11 CoverageInstruction

Role Name	Name	Description	Mult.	Type
Class	CoverageInstruction	A drawing instruction for depicting coverages of data	-	-
Role	coverageFill	The coverage fill used for depiction	1	Coverages::CoverageFill

9-11.2.12 AugmentedGeometry

Role Name	Name	Description	Mult.	Type
Class	AugmentedGeometry	A base class for drawing instructions that uses geometry not available in the dataset. The geometry is generated by the portrayal functions according to a defined CRS	-	-

Attribute	crs	The coordinate reference system of the generated geometry. One of • Geographic CRS • Portrayal CRS • Local CRS For detailed description see the documentation of the GraphicsBase package	1	GraphicBase::CRSType
Role	text	A text to be depicted by the instruction. The rules for text apply depending on the type of geometry used by the instruction	0..1	Text::Text

9-11.2.13 AugmentedPoint

Role Name	Name	Description	Mult.	Type
Class	AugmentedPoint	A drawing instruction for a point symbol where the position is not given by the feature type	-	-
Attribute	position	The position of the symbol	1	GraphicBase::Point
Role	symbol	The symbol to be depicted	0..1	Symbol::Symbol

9-11.2.14 AugmentedLineOrArea

Role Name	Name	Description	Mult.	Type
Class	AugmentedLineOrArea	A base class for linear augmented geometry	-	-
Role	lineStyle	The line style to be depicted	0..1	LineStyles::LineStyle

9-11.2.15 AugmentedRay

Role Name	Name	Description	Mult.	Type
Class	AugmentedRay	A drawing instruction that defines a line from the position of a point feature to another position. The position is defined by the direction and the length attributes. It can be used for drawing line styles or line texts	-	-
Attribute	rotationCRS	If present, specifies the CRS for *direction*	0..1	GraphicsBase::CRSType
Attribute	direction	The direction of the ray relative to the used CRS	1	double
Attribute	length	The length of the ray. The units depending on the used CRS	1	double

9-11.2.16 AugmentedPath

Role Name	Name	Description	Mult.	Type
Class	AugmentedPath	A drawing instruction for a line. It can be used for drawing line styles or line texts	-	-
Role	path	The path defining the line geometry	1	GraphicsBase::Path

9-11.2.17 AugmentedArea

Role Name	Name	Description	Mult.	Type
Class	AugmentedArea	A drawing instruction for an area. It can be used for drawing line styles, area fills, or area texts. The used path must be closed	-	-
Role	areaFill	The area fill to be depicted	0..1	AreaFills::AreaFill

9-12 Symbol Definitions

9-12.1 Overview

The SymbolDefinition package describes the graphic primitives used for the portrayal. Parts of the primitives are defined externally by using SVG definitions. Those external parts will be referenced from the types in this model. The package diagram is shown in the Figure 9-11 below.

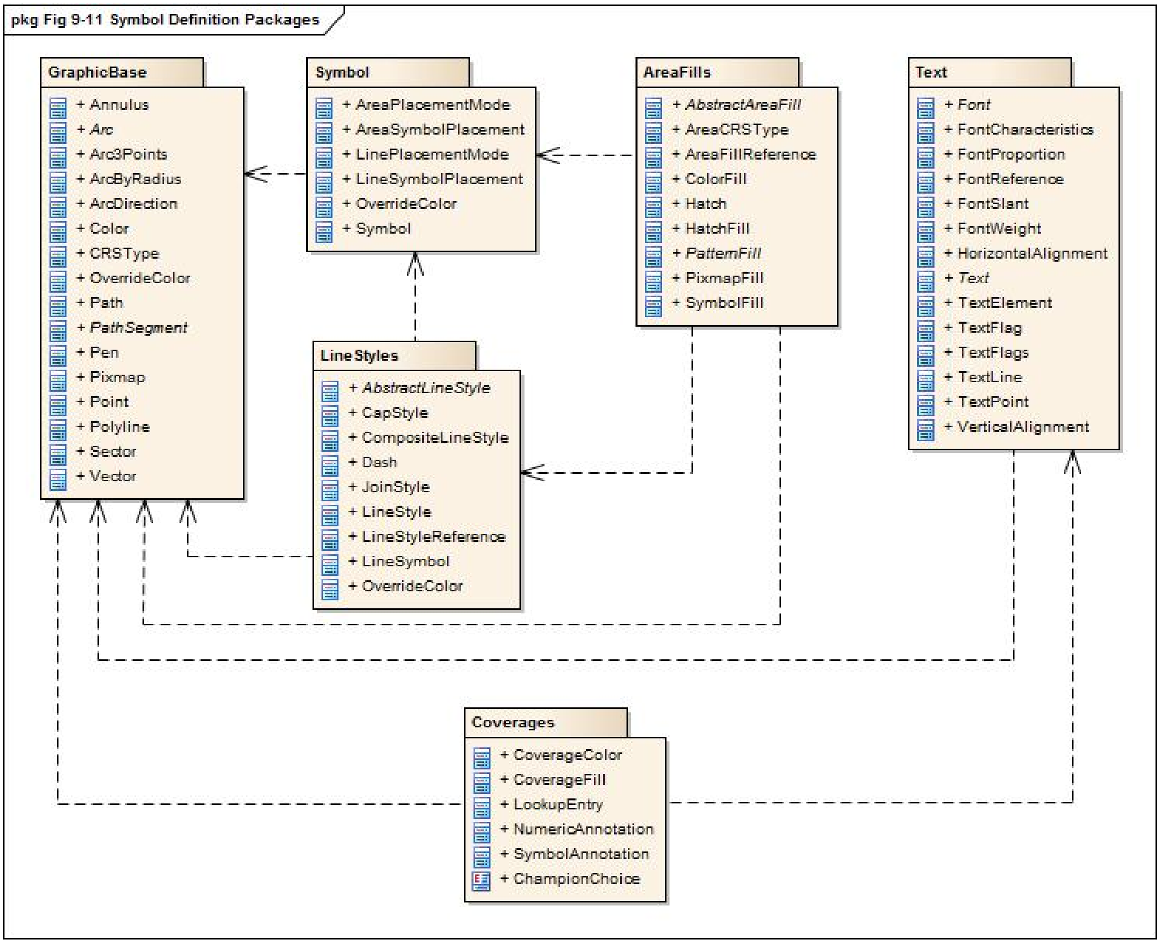


Figure 9-11 — Symbol Definition Packages

9-12.2 The GraphicBase package

9-12.2.1 Overview

This package contains graphic base types for the use in other packages.

9-12.2.2 Model

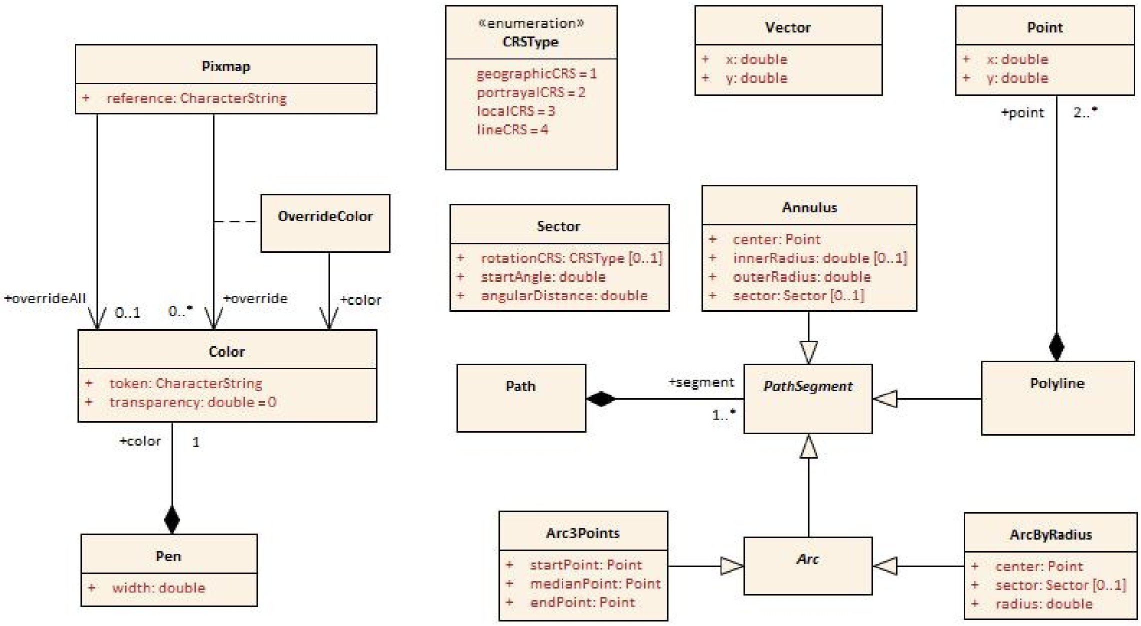


Figure 9-12 — Graphics Base

9-12.2.2.1 Point

Role Name	Name	Description	Mult.	Type
Class	Point	A zero-dimensional geometric object in a two-dimensional coordinate space. The coordinate will refer to a coordinate reference system	-	-
Attribute	x	The x-coordinate of the point. In case the CRS is a geographic CRS this refers to the longitude	1	double
Attribute	y	The y-coordinate of the point. In case the CRS is a geographic CRS this refers to the latitude	1	double

9-12.2.2.2 Vector

Role Name	Name	Description	Mult.	Type
Class	Vector	A geometric object that has both a magnitude and a direction. It is limited to Cartesian coordinate reference systems	-	-
Attribute	x	The x-coordinate of the vector	1	double
Attribute	y	The y-coordinate of the vector	1	double

9-12.2.2.3 Color

Role Name	Name	Description	Mult.	Type
Class	Color	Representing a colour according to the colour model	-	-
Attribute	token	The token specifies either an element in a colour table or a colour definition in the RGB space	1	string

Attribute	transparency	The value specifies the transparency; between 0 (opaque) and 1 (full transparent)	1	double

9-12.2.2.4 Pen

Role Name	Name	Description	Mult.	Type
Class	Pen	A tool for drawing lines	-	-
Attribute	width	The width of the pen in mm	1	double
Role	color	The colour of the pen comprises the actual colour and the transparency	1	Color

9-12.2.2.5 Pixmap

Role Name	Name	Description	Mult.	Type
Class	Pixmap	A two dimensional array of pixels defining an image	-	-
Attribute	reference	A reference to an external definition of the pixmap. This string is a unique identifier within the pixmap section of the Portrayal Catalogue	1	string
Role	overrideAll	A colour that override all none fully transparent colours used within the pixmap	0..1	Color
Association	override	A colour to be replaced by another colour	0..*	OverrideColor

9-12.2.2.6 OverrideColor

Role Name	Name	Description	Mult.	Type
Class	OverrideColor	Association class for the replacement of an existing colour in the pixmap with another colour	-	-
Role	color	The colour that is used to replace the existing colour in the pixmap	1	Color

9-12.2.2.7 CRSType

Role Name	Name	Description
Type	CRSType	The value describes the type of a CRS. This includes the axes definitions, base line for angle measurement and units for distances
Enumeration	geographicCRS	A geographic CRS with axis latitude and longitude measured in degrees. Angles are defined clockwise from the true north direction. Distances will be measured in metres
Enumeration	portrayalCRS	A Cartesian coordinate system with the y-axis pointing upwards. Units on the axes and for distances are millimetres. Angles are measured in degrees clockwise from the positive y-axis. Note that the actual output device may have a different orientation of the y-axis
Enumeration	localCRS	A Cartesian coordinate system originated at a local geometry. Units on the axes and for distances are millimetres. Angles are measured in degrees clockwise from the positive y-axis. See explanations for details
Enumeration	lineCRS	A none-Cartesian coordinate system where the x-axis is following the geometry of a curve and the y-axis is perpendicular to the x-axis (positive to the left of the x-axis). Units on the axes and for distances are millimetres. Angles are measured in degrees clockwise from the positive y-axis

Figure 9-13 below shows how the local CRS are defined for the different types of geometry.

From left to right:

- Local CRS for point geometry
  Note: for multi points the local CRS is repeated at each point.

- Local CRS for curve geometry. The origin of the coordinate system can be any point of the line. This point can be defined by the absolute or relative distance from the start of the line. The x-axis is directed in the direction of the tangent at the tangency point and the y-axis is oriented perpendicular to this direction.
- Local CRS for surface geometry. For the boundary the same rules apply as for curve geometry. For the interior of the surface a coordinate system is used that has axes parallel to the Portrayal CRS. The origin can be an arbitrary point that is constant relative to the surface. This point can be outside the surface.

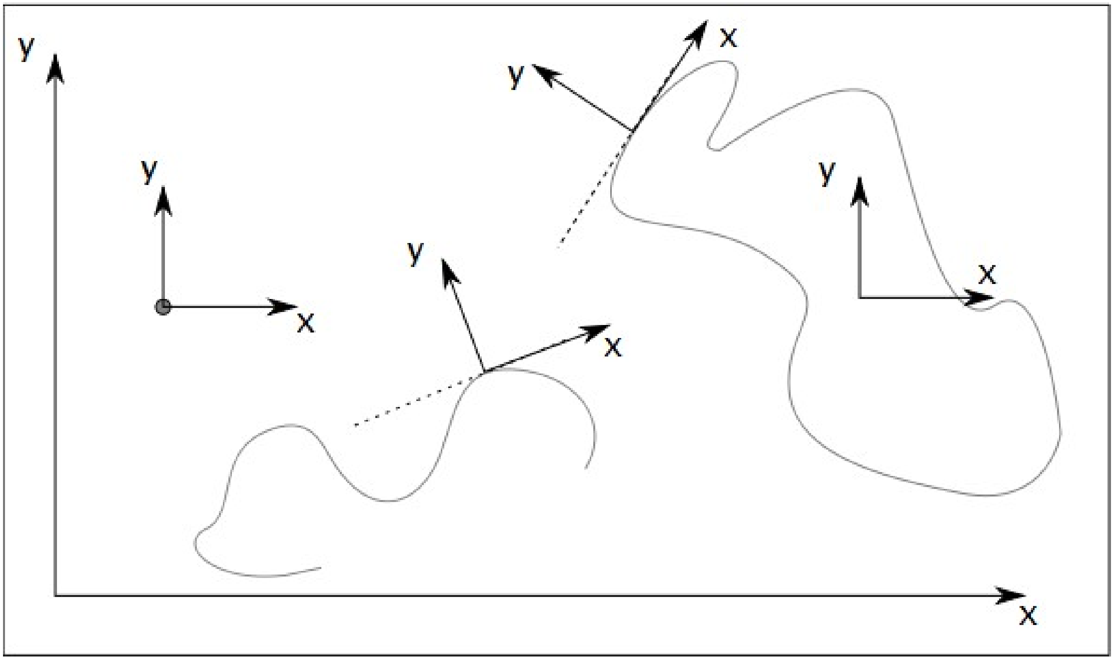


Figure 9-13 — Local CRS

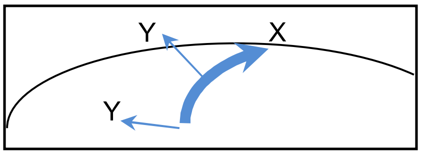


Figure 9-14 — Line CRS

9-12.2.2.8 Sector

Role Name	Name	Description	Mult.	Type
Class	Sector	Region of the Cartesian plane enclosed by two radii	-	-
Attribute	rotationCRS	If present, specifies the CRS for startAngle	0..1	CRSType
Attribute	startAngle	The direction of the radius that defines the beginning of the sector	1	double
Attribute	angularDistance	The angular distance of the sector measured in degrees. Positive values means clockwise, negative values means anti-clockwise	1	double

9-12.2.2.9 Path

Role Name	Name	Description	Mult.	Type
Class	Path	The definition of linear geometry by a composition of segments	-	-

Role	segment	The segments that build up the path	1..*	PathSegment

Paths can be closed or not closed. A closed path has coinciding start and end points. Segments are connected until a path is closed. In that case the next segment is not connected and the path contains multiple sub-paths.

9-12.2.2.10 PathSegment

Role Name	Name	Description	Mult.	Type
Class	PathSegment	Abstract base class for all segments that can be used within a path	-	-

9-12.2.2.11 Polyline

Role Name	Name	Description	Mult.	Type
Class	Polyline	A segment defining its geometry by a series of points	-	-
Role	point	The segments the build up the path	2..*	Point

9-12.2.2.12 Arc

Role Name	Name	Description	Mult.	Type
Class	Arc	Abstract base class for segments describing arcs of a circle	-	-

9-12.2.2.13 Arc3Points

Role Name	Name	Description	Mult.	Type
Class	Arc3Points	A segment describing an arc of a circle that is defined by 3 points. The points must not be colinear	-	-
Attribute	startPoint	The point where the arc starts	1	Point
Attribute	medianPoint	An arbitrary point on the arc	1	Point
Attribute	endPoint	The point where the arc ends	1	Point

9-12.2.2.14 ArcByRadius

Role Name	Name	Description	Mult.	Type
Class	ArcByRadius	A segment describing an arc of a circle that is defined by the centre of the arc and a radius. Optional the arc can be restricted by a sector	-	-
Attribute	center	The centre of the arc	1	Point
Attribute	sector	The sector defining where the arc starts and end. If not present the arc is a full circle	0..1	Sector
Attribute	radius	The radius of the circle	1	double

9-12.2.2.15 Annulus

Role Name	Name	Description	Mult.	Type
Class	Annulus	A ring-shaped region bounded by two concentric circles. Optionally it can be enclosed by two radii of the circle	-	-
Attribute	center	The centre of the arc	1	Point

Attribute	innerRadius	The radius of the smaller circle. If not present the segment describes a sector of a circle	0..1	double
Attribute	outerRadius	The radius of the larger circle	1	double
Attribute	sector	The sector of an annulus segment	0..1	Sector

9-12.3 The Symbol package

9-12.3.1 Model

This package contains the model of a symbol. Note that the definition of the symbol graphic itself is not the subject of this model. This will be defined in external files according to the SVG 1.1 recommendation.

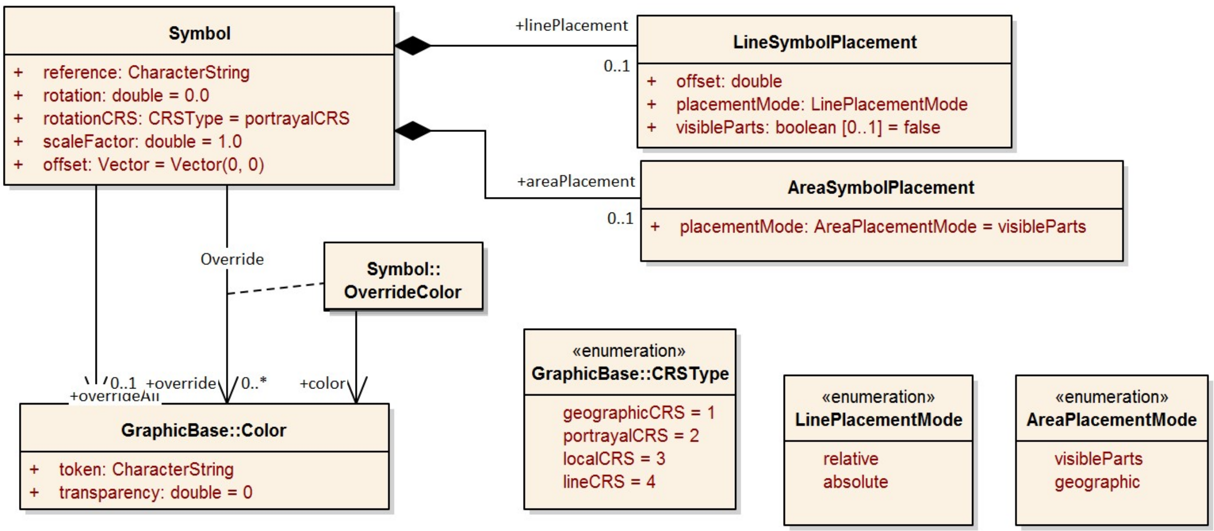


Figure 9-15 — Symbol Package

9-12.3.1.1 Symbol

Role Name	Name	Description	Mult.	Type
Class	Symbol	A two dimensional graphical element	-	-
Attribute	reference	A reference to an external definition of the symbol graphic. This is an unique identifier in the symbol section of the Portrayal Catalogue	1	string
Attribute	rotation	The rotation angle of the symbol. The default value is 0	1	double
Attribute	rotationCRS	Specifies the coordinate reference system for the rotation	1	GraphicsBase::CRSTyp e
Attribute	scaleFactor	The factor by which the original symbol graphic is scaled. The default value is 1	1	double
Attribute	offset	The shift of the symbols position from the position of the geometry. The default value is the vector with length equals to 0	1	GraphicsBase::Vector
Role	overrideAll	A colour that override all none fully transparent colours used within the symbol	0..1	GraphicsBase::Color
Association	override	A colour to be replaced by another colour	0..*	OverrideColor
Role	linePlacement	Information where on a line the symbol should be placed	0..1	LineSymbolPlacement
Role	areaPlacement	Defines the placement of a symbol within an area	0..1	AreaSymbolPlacement

9-12.3.1.2 OverrideColor

Role Name	Name	Description	Mult.	Type
Class	OverrideColor	Association class for the replacement of an existing colour in the symbol	-	-
Role	color	The colour that is used to replace an existing colour in the symbol	1	Color

9-12.3.1.3 LineSymbolPlacement

Role Name	Name	Description	Mult.	Type
Class	LineSymbolPlacement	Defines the placement of a symbol along a line	-	-
Attribute	offset	The offset from the start of the curve	1	double
Attribute	placementMode	The mode that defines how the offset is to be interpreted	1	LinePlacementMode
Attribute	visibleParts	When true, indicates the placement applies to each visible part of the curve. Default = false	0..1	boolean

9-12.3.1.4 AreaSymbolPlacement

Role Name	Name	Description	Mult.	Type
Class	AreaSymbolPlacement	Defines the placement of a symbol within an area	-	-
Attribute	placementMode	The mode that defines how the symbol has to placed.	1	AreaPlacementMode

9-12.3.1.5 LinePlacementMode

Role Name	Name	Description
Type	LinePlacementMode	Defines the type of placement of a symbol along a line
Enumeration	relative	The offset has to be interpreted as homogenous coordinates, 0 for the start and 1 for the end of the curve
Enumeration	absolute	The offset is the distance from the start of the curve

9-12.3.1.6 AreaPlacementMode

Role Name	Name	Description
Type	AreaPlacementMode	Defines the type of placement of a symbol within an area
Enumeration	visibleParts	The symbol has to be placed at a representative position in each visible part of the surface
Enumeration	geographic	The symbol has to be placed at a representative position of the geographic object

9-12.4 The LineStyles package

9-12.4.1 Model

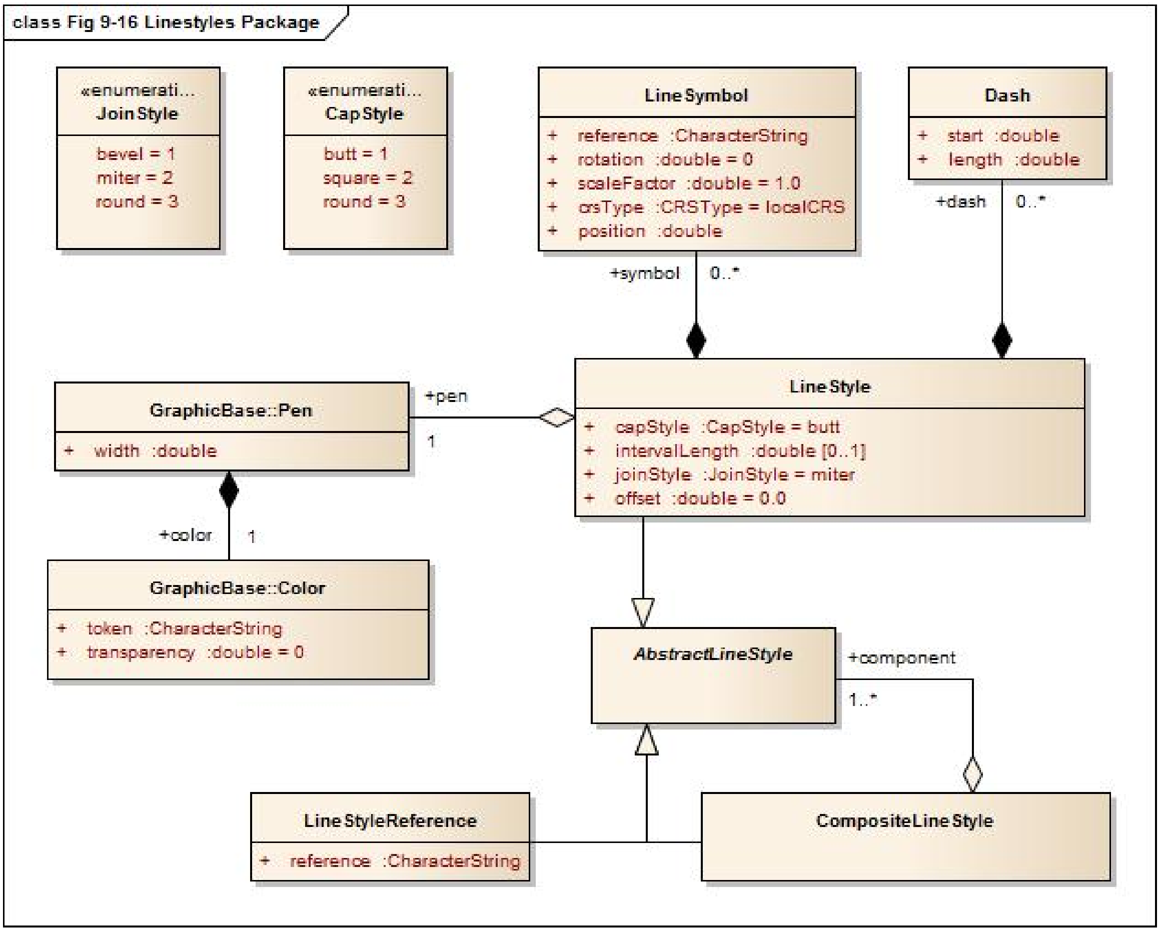


Figure 9-16 — Line Styles Package

9-12.4.1.1 AbstractLineStyle

Role Name	Name	Description	Mult.	Type
Class	AbstractLineStyle	Abstract base class for graphics to depict line geometry	-	-

9-12.4.1.2 LineStyle

Role Name	Name	Description	Mult.	Type
Class	LineStyle	A style for line geometry either solid or dashed	-	-
Attribute	offset	An offset perpendicular to the direction of the line. The value refers to the y-axis of the line CRS (positive to the left, mm)	1	double
Attribute	capStyle	The decoration that is applied where a line segment ends	1	CapStyle
Attribute	joinStyle	The decoration that is applied where two line segments meet	1	JoinStyle
Attribute	intervalLength	The length of a repeating interval of the line style along the x-axis of the line CRS (units in mm) If not defined the line style describes a solid line	0..1	double
Role	dash	The dashes of a dashed line style	0..*	Dash
Role	pen	The pen used for drawing the line	1	Pen

Role	symbol	Symbols placed along the line	0..*	LineSymbol

9-12.4.1.3 Dash

Role Name	Name	Description	Mult.	Type
Class	Dash	A single dash in a repeating line pattern	-	-
Attribute	start	The start of the dash measured from the start of the repeating interval,along the x-axis of the line CRS (units in mm)	1	double
Attribute	length	The length of the dash along the x-axis of the line CRS (units in mm)	1	double

9-12.4.1.4 LineSymbol

Role Name	Name	Description	Mult.	Type
Class	LineSymbol	A symbol placed along a line in a repeating pattern	-	-
Attribute	reference	A reference to an external definition of the symbol graphic. This refers to an identifier of a Catalogue item	1	string
Attribute	rotation	The rotation angle of the symbol. The default value is 0	1	double
Attribute	scaleFactor	The scale factor of the symbol. The default is 1.0	1	double
Attribute	crsType	The type of the CRS where the symbol has to be transformed to. Possible values are localCRS, lineCRS and portrayalCRS	1	CRSType
Attribute	position	The position of the symbol measured from the start of the repeating interval,along the x-axis of the line CRS (units in mm)	1	double

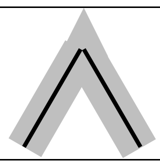
9-12.4.1.5 CompositeLineStyle

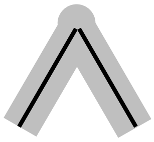
Role Name	Name	Description	Mult.	Type
Class	CompositeLineStyle	A line style made with an aggregation of other line styles		
Role	component	The components of the composite line style	1..*	AbstractLineStyle

9-12.4.1.6 LineStyleReference

Role Name	Name	Description	Mult.	Type
Class	LineStyleReference	A line style defined in an external file		
Attribute	reference	The reference to the external definition of the line style. This is an unique identifier in the line style section of the Portrayal Catalogue	1	string

9-12.4.1.7 JoinStyle

Role Name	Name	Description
Type	JoinStyle	The decoration that is applied where two line segments meet
Enumeration	bevel	
Enumeration	miter	

Enumeration	round	

9-12.4.1.8 CapStyle

Role Name	Name	Description
Type	CapStyle	The decoration that is applied where a line segment ends.
Enumeration	butt	
Enumeration	square	
Enumeration	round	

9-12.5 The AreaFills package

9-12.5.1 Model

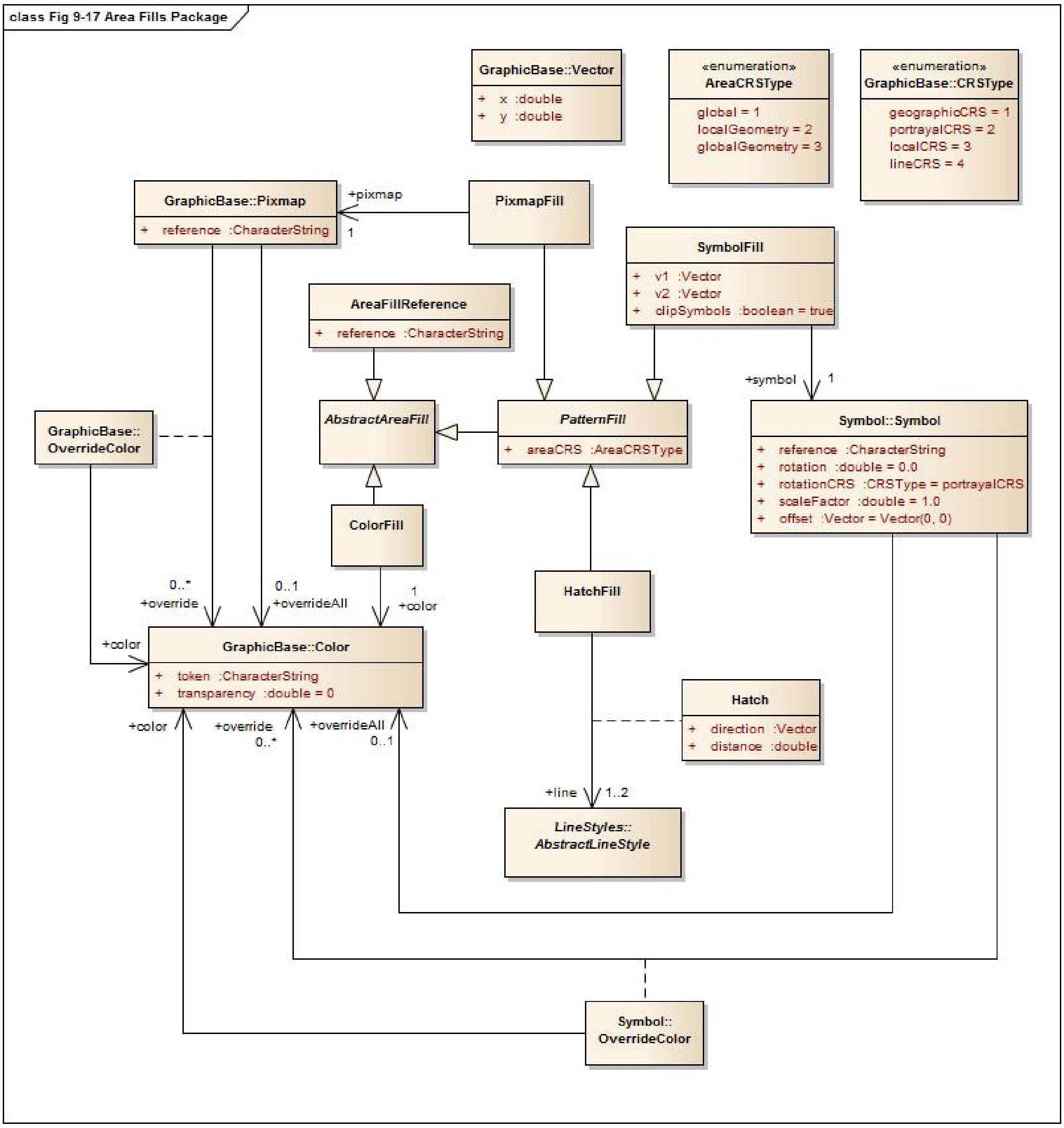


Figure 9-17 — Area Fills Package

9-12.5.1.1 AbstractAreaFill

Role Name	Name	Description	Mult.	Type
Class	AbstractAreaFill	Abstract base class for graphics that are designed to fill an area	-	-

9-12.5.1.2 PatternFill

Role Name	Name	Description	Mult.	Type
Class	PatternFill	Abstract base class for pattern area fills	-	-
Attribute	areaCRS	Coordinate reference system which defines the origin of the pattern	1	AreaCRSType

9-12.5.1.3 AreaFillReference

Role Name	Name	Description	Mult.	Type
Class	AreaFillReference	An area fill defined in an external file	-	-
Attribute	reference	The reference to the external definition. This is an unique identifier in the area fill section of the Portrayal Catalogue	1	string

9-12.5.1.4 ColorFill

Role Name	Name	Description	Mult.	Type
Class	ColorFill	Class defining a solid colour fill for an area	-	-
Role	color	References the colour and transparency for the colour fill	1	Color

9-12.5.1.5 PixmapFill

Role Name	Name	Description	Mult.	Type
Class	PixmapFill	Pattern fill where the pattern is defined by a pixmap	-	-
Role	pixmap	The pixmap defining the pattern	1	Pixmap

9-12.5.1.6 SymbolFill

Role Name	Name	Description	Mult.	Type
Class	SymbolFill	Pattern fill where the pattern is defined by repeated symbols	-	-
Role	symbol	The symbol used for the pattern	1	Symbol
Attribute	v1	Defines the offset of the next symbol in the first dimension of the pattern according to the local CRS	1	Vector
Attribute	v2	Defines the offset of the next symbol in the second dimension of the pattern according to the local CRS	1	Vector
Attribute	clipSymbols	Indicates whether the symbols in the pattern are to be clipped by the area (when they are part in/out of the area) or whether the symbol is not drawn at all unless it is completely contained in the area	1	boolean *True*: Fill symbols are clipped at area boundaries *False*: Fill symbols extending over the area boundaries are not drawn at all

9-12.5.1.7 HatchFill

Role Name	Name	Description	Mult.	Type
Class	HatchFill	Defining a pattern made of one or two sets of parallel lines	-	-
Association	Hatch	A set of parallel lines	1	Hatch

9-12.5.1.8 Hatch

Role Name	Name	Description	Mult.	Type
Class	Hatch	A set of parallel lines used for an area fill pattern		
Attribute	direction	The vector defining the direction of the set of lines	1	Vector

Attribute	distance	The distance between the lines measured perpendicular to the direction	1	double
Role	line	The line style used for each hatch line	1..2	LineStyles:: AbstractLineStyle

9-12.5.1.9 AreaCRSType

Role Name	Name	Description
Type	PatternCRS	Describes how a fill patter is referenced
Enumeration	global	Anchor point is consistent with a location on the drawing device for, example starting with the corner of the screen. As screen pans the pattern will appear to shift/move through the object on screen
Enumeration	localGeometry	Anchor point is consistent with the local geometry of the object being depicted, for example the upper left corner of the object. Patterns of adjacent objects may not match
Enumeration	globalGeometry	The anchor point of the fill pattern is defined at a common location such that patterns remain consistent relative to all area objects

9-12.6 The Text package

9-12.6.1 Overview

The text package contains the types necessary for the depiction of text. This includes fonts. In this model fonts may be described by characteristics or referenced by name. Two types of text instructions are supported:

- Text relative to a point
- Text that will be drawn along a linear geometry

9-12.6.2 Fonts

A font is a set of typefaces. A typeface is the artistic representation or interpretation of characters; it is the way the type looks.

This standard supports two methods of defining fonts, the first describes a font by four attributes and let the system find a best match to an actual font available on the graphic system. The second method is referencing an external font file. The format of this file must conform to the 'True Type Font' standard and must be included in the the Portrayal Catalogue.

9-12.6.3 Model

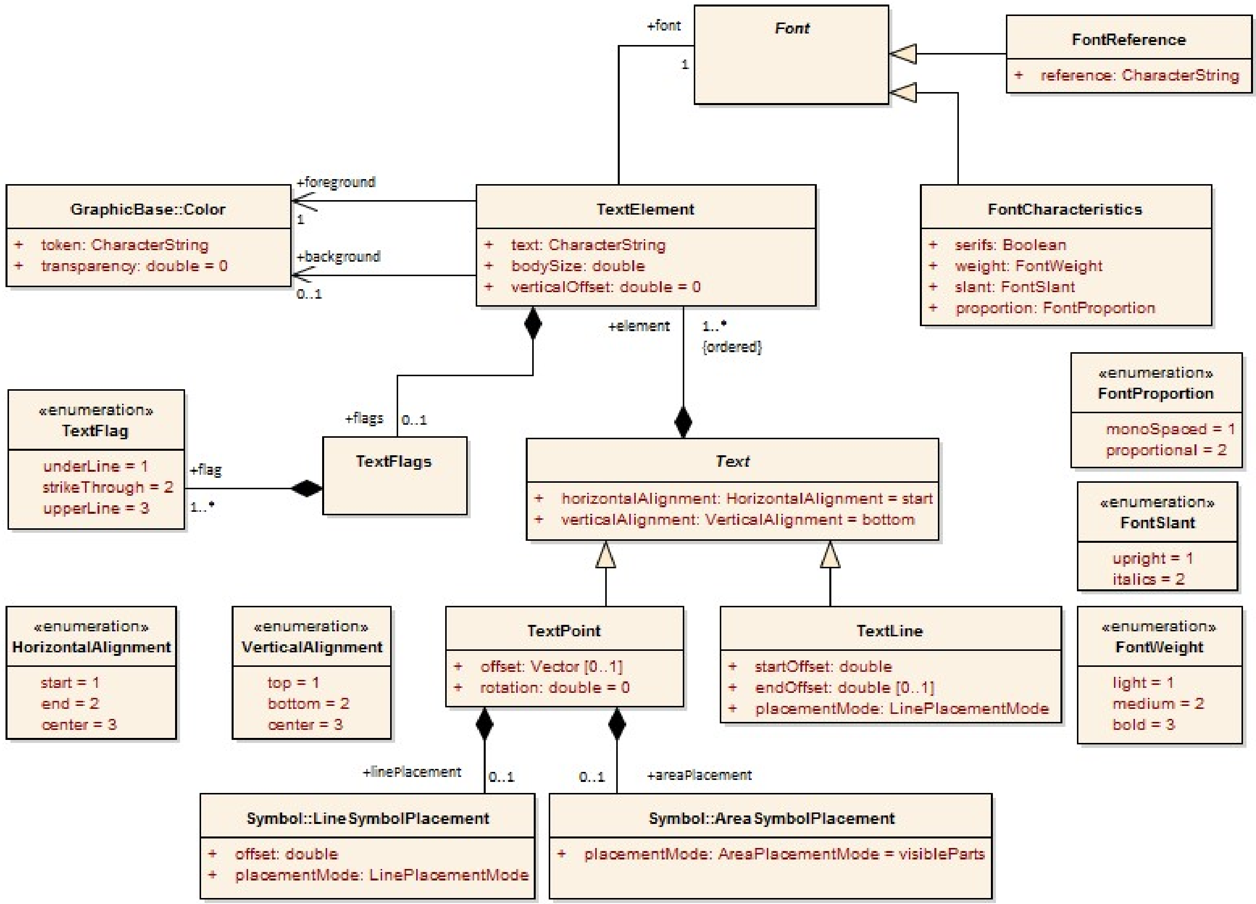


Figure 9-18 — Text Package

9-12.6.3.1 Font

Role Name	Name	Description	Mult.	Type
Class	Font	Abstract base class for fonts	-	-

9-12.6.3.2 FontCharacteristics

Role Name	Name	Description	Mult.	Type
Class	FontCharacteristics	Class describing the main characteristics of a font	-	-
Attribute	serifs	Describes whether the typefaces contain serifs or not	1	boolean
Attribute	weight	Describes the thickness of the typefaces	1	FontWeight
Attribute	slant	Describes the slant of the typefaces	1	FontSlant
Attribute	proportion	Describes whether all typefaces in the font have an individual width or a fixed width	1	FontProportion

9-12.6.3.3 FontReference

Role Name	Name	Description	Mult.	Type
Class	FontReference	Class referencing a font from an external source	-	-
Attribute	reference	The identifier for the external file within the Portrayal Catalogue	1	string

9-12.6.3.4 Text

Role Name	Name	Description	Mult.	Type
Class	Text	The abstract base class of graphic elements for depicting text. The text is composed of elements	-	-
Attribute	horizontalAlignment	Specifies how the text is horizontally aligned relative to the anchor point. Default = start	1	HorizontalAlignment
Attribute	verticalAlignment	Specifies how the text is vertically aligned relative to the anchor point. Default = bottom	1	VerticalAlignment
Role	element	The ordered list of text elements	1..*	TextElement

9-12.6.3.5 TextPoint

Role Name	Name	Description	Mult.	Type
Class	TextPoint	A graphic element for depicting text relative to a point	-	-
Attribute	offset	Specifies the offset from the anchor point with respect to the portrayal CRS	0..1	GraphicsBase::Vector
Attribute	rotation	Specifies the rotation angle relative to the portrayal CRS. Default = 0	1	double
Role	linePlacement	Describes the placement of the text when the geometry is a curve	0..1	Symbol:: LineSymbolPlacement
Role	areaPlacement	Describes the placement of the text when the geometry is a surface	0..1	Symbol:: AreaSymbolPlacement

9-12.6.3.6 TextLine

Role Name	Name	Description	Mult.	Type
Class	TextLine	A graphic element for depicting text along linear geometry	-	-
Attribute	startOffset	This offset specifies the anchor point on the line	1	double
Attribute	endOffset	This offset specifies the stop point of the text at the line. If present the startOffset does not specifies an anchor point but the start point of the text. The text will evenly be spaced between the two positions. Horizontal alignment has no effect in this case	0..1	double

Attribute	placementMode	Specifies how the offsets have to be interpreted	1	Symbol:: LinePlacementMode

9-12.6.3.7 TextFlags

Role Name	Name	Description	Mult.	Type
Class	TextFlags	A container for text flags	-	-
Role	flag	A text flag	1..*	TextFlag

9-12.6.3.8 TextElement

Role Name	Name	Description	Mult.	Type
Class	TextElement	A sub element of a graphic text	-	-
Attribute	text	The text to be depicted	1	string
Attribute	bodySize	This property describes the size with which the text will be depicted	1	double
Attribute	verticalOffset	The vertical offset in mm between the base line of the text element and the base line of the text. This can be used to generate sub- or superscripts. Default = 0	1	double
Role	flags	Flags describe special properties of the text element like underline etc.	0..1	TextFlags
Role	font	The font used for the depiction of the text element.	1	Font
Role	foreground	The colour used to depict the glyphs	1	Color
Role	background	The colour to fill the rectangle surrounding the text element before the text is depicted. If not given there is no fill (transparent)	0..1	Color

9-12.6.3.9 FontSlant

Role Name	Name	Description
Type	FontSlant	The slant used within a font
Enumeration	upright	Typefaces are upright
Enumeration	italics	Typefaces are cursive

9-12.6.3.10 FontWeight

Role Name	Name	Description
Type	FontWeigth	The thickness used for the typefaces in a font
Enumeration	light	Typefaces are depicted as thin (standard)
Enumeration	medium	Typefaces are depicted thicker as 'Light' but not as thin as 'Bold'
Enumeration	bold	Typefaces are depicted more prominent (**Bold**)

9-12.6.3.11 FontProportion

Role Name	Name	Description
Type	FontProportion	The values describe how the width of the typefaces in a font is defined
Enumeration	monoSpaces	All typefaces in a font have the same width, also known as 'typewriter' fonts
Enumeration	proportional	Any typeface in the font as its individual width

9-12.6.3.12 TextFlag

Role Name	Name	Description
Type	TextFlag	The values describe some effects used when the text will be depicted. The values can be combined
Enumeration	underLine	Text is depicted with a line under the text
Enumeration	strikeThrough	Text is depicted struck through, a line goes through the text
Enumeration	upperLine	Text is depicted with a line above the text

9-12.6.3.13 VerticalAlignment

Role Name	Name	Description
Type	VerticalAlignment	Describes the text placement relative to the anchor point in vertical direction
Enumeration	top	The anchor point is at the top of the text
Enumeration	bottom	The anchor point is at the bottom of the text
Enumeration	center	The anchor point is at the (vertical) centre of the text

9-12.6.3.14 HorizontalAlignment

Role Name	Name	Description
Type	HorizontalAlignment	Describes the text placement relative to the anchor point in horizontal direction
Enumeration	start	The anchor point is at the start of the text
Enumeration	end	The anchor point is at the end of the text
Enumeration	center	The anchor point is at the (horizontal) centre of the text

9-12.7 The Coverage package

9-12.7.1 Overview

The coverage package contains the types for the depiction of a Coverage. This portrayal is applicable to the portrayal of numeric Coverage values. Three types of coverage portrayals are supported:

- Colour;
- Numeric Annotation; and
- Symbol Annotation.

9-12.7.2 Ranges

Ranges are used to control how portrayal is assigned to the values in a Coverage. These make use of the S-100_NumericRange complex type which is defined in S-100 Part 1 Conceptual Schema Language. The Numeric Range type allows for various range definitions with different closure options.

9-12.7.3 Lookup Table

The CoverageFill class carries an ordered list of lookup entries. Each of these entries carries a range used to evaluate a match by testing if the coverage value matches the range. The first lookup entry with a matching range is used to apply up to one of each type of portrayal (colour, numeric annotation or a symbol) to the coverage element. This allows for example to fill a cell in a grid with a colour and assign a numeric or symbol annotation to the cell as well.

The lookup table can also associate an alert specified in the drawing instruction with a range of coverage values. When associating alerts with coverage values there may or may not be portrayal elements (colour, numeric annotation or a symbol) present.

9-12.7.4 Model

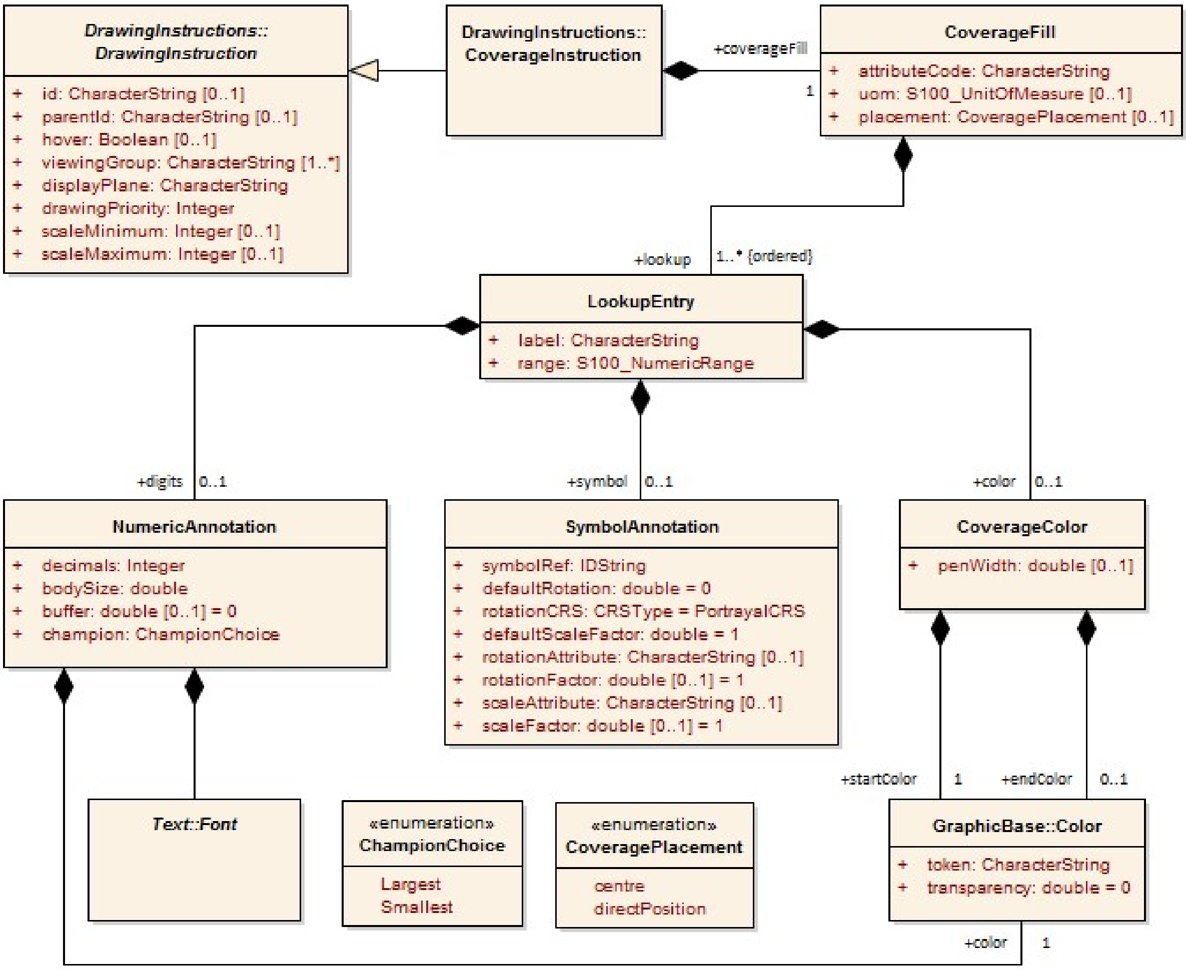


Figure 9-19 — Coverage Package

9-12.7.4.1 CoverageFill

Role Name	Name	Description	Mult.	Type
Class	CoverageFill	A class to fill a Coverage with using a lookup table to match a value or range of values and assign colour, numeric or symbol annotations	-	-
Attribute	attributeCode	Code of coverage attribute value to match	1	CharacterString
Attribute	uom	Unit of measure. If not given the values in the range are assumed to be same units as the coverage attribute values	0..1	S100_UnitOfMeasure
Attribute	placement	Designation of anchor point for placement of text or symbol annotation Mandatory if and only if portrayal requires text or symbol annotation; otherwise omitted This attribute is valid only for continuous coverages (for discrete coverages, the anchor point is the direct position). See clause 9-12.7.4.7	0..1	CoveragePlacement
Role	lookup	Lookup table. The entries are ordered and processed on a first match basis	1..*	LookupEntry

9-12.7.4.2 LookupEntry

Role Name	Name	Description	Mult.	Type
Class	LookupEntry	An entry in a lookup table used to assign portrayal to coverage elements.	-	-
Attribute	label	String used as a display label or legend field.	1	CharacterString
Attribute	range	Value range definition. Can be a single value, open or closed range etc. See S-100 Part 1 Conceptual Schema Language for details.	1	S100_NumericRange
Role	color	The colour to assign to the matching range. Can be a single colour or a colour ramp.	0..1	CoverageColor
Role	digits	Display the value as numeric digits.	0..1	NumericAnnotation
Role	symbol	Display a symbol.	0..1	SymbolAnnotation

9-12.7.4.3 CoverageColor

Role Name	Name	Description	Mult.	Type
Class	CoverageColor	A class to fill a Coverage with colour	-	-
Attribute	penWidth	Optional pen width to apply for dot colour used for discrete points	0..1	double
Role	startColor	The colour to assign to the matching range or to use as the start point in a colour ramp when 'endColor' is defined	1	GraphicBase::Color
Role	endColor	The colour to use as stop point in a colour ramp. The range of values is spread linearly across the range of colours from 'startColor' to 'endColor' to produce a gradient effect The start and end colours shall be converted to the CIE xyL colour space and each component (that is x, y, L and transparancy) shall be interpolated individually before converting back to sRGB for display	0..1	GraphicBase::Color

9-12.7.4.4 NumericAnnotation

Role Name	Name	Description	Mult.	Type
Class	NumericAnnotation	A class for numeric textual annotations of values in a Coverage	-	-
Attribute	decimals	Number of decimal digits to show in subscript	1	integer
Attribute	bodySize	This property describes the size with which the text will be depicted	1	double
Attribute	buffer	Buffer to apply for collision detection in presentation units. Default=0	1	double
Attribute	champion	Enumeration to indicate which value to display in the event of a collision	1	ChampionChoice
Role	font	Font information to use for display of numeric values across a coverage. Text::Font is a choice of either FontCharacteristics or FontReference	1	Text::Font
Role	color	Colour to draw the numeric annotation	1	GraphicBase::Color

9-12.7.4.5 SymbolAnnotation

Role Name	Name	Description	Mult.	Type
Class	SymbolAnnotation	A class for symbol annotations of values in a coverage.	-	-
Attribute	symbolRef	Reference to the symbol to apply. Catalogue id.	1	IDString

Attribute	defaultRotation	A default symbol rotation. Applies when rotation attribute not defined. Default=0	0..1	double
Attribute	rotationCRS	Specifies the coordinate reference system for the rotation. Default=PortrayalCRS	1	GraphicsBase::CRSType
Attribute	defaultScale	A default symbol scale factor. Applies when scale attribute not defined. Default=1	1	double
Attribute	rotationAttribute	The attribute code of the Coverage Attribute to use for the symbol rotation value.	0..1	CharacterString
Attribute	rotationFactor	Used to adjust the 'rotationAttribute' value by multiplication before applying. Default 1.0	0..1	double
Attribute	scaleAttribute	The attribute code of the Coverage attribute to use for scaling the symbol size.	0..1	CharacterString
Attribute	scaleFactor	Used to adjust the ' scaleAttribute' value by multiplication before applying. Default 1.0	0..1	double

9-12.7.4.6 CoveragePlacement

Role Name	Name	Description
Type	CoveragePlacement	Defines the type of placement of a symbol or text annotations for a coverage
Enumeration	centre	The anchor point for annotations is the centre of the cell
Enumeration	directPosition	The anchor point for annotations is the direct position associated with each value of the attribute designated by the attributeCode parameter of the CoverageFill in which the placement is encoded

9-12.7.4.7 Coverages and placement

Table 9-1 Coverages and placement

Coverage spatial type (attribute dataCodingFormat). See clause 10c-10.1 Data coding format	Interpolation (attribute interpolationType). See clause 8-7.7.3 S100_CV_InterpolationMethod	placement See clauses 9-12.7.4.1 CoverageFill and 9-12.7.4.6 CoveragePlacement	Resulting symbol/text placement
fixedStations movingPlatform stationwiseFixed Feature oriented Regular Grid (DCF 1, 4, 8, 9)	(not allowed)	(Not used. Ignore if encoded)	each direct position
regularGrid ungeorectifiedGrid irregularGrid (DCF 2, 3, 5)	discrete	directPosition OR (not present)	each direct position
		centre	ERROR - ignore placement attribute and portray at each direct position
	(other)	directPosition	each direct position
		centre OR (not present)	centre of each cell
variableCellSize (DCF 6)	discrete	directPosition OR (not present)	each direct position for each unit grid cell
		centre	ERROR - ignore placement attribute and portray at each direct position

	(other)	directPosition	each direct position for each unit grid cell
		centre OR (not present)	centre of each unit grid cell
TIN (DCF 7)	discrete	directPosition OR (not present)	each direct position (triangle vertex)
		centre	ERROR - ignore placement attribute and portray at each direct position (triangle vertex)
	(other)	directPosition	each direct position (triangle vertex)
		centre OR (not present)	barycentre (centroid) of each triangle

9-13 The portrayal library

9-13.1 Overview

- Machine readable.
- A file/directory structure with a Catalogue file.
- Files for pixmaps, symbols, complex line styles, areafills, fonts and colour profiles.
- Alert information in a separate file.
- Portrayal rules in separate files.
- Model and Schema for the Catalogue included.

9-13.2 Structure

```
Root ---- (contains the Catalogue named "portrayal_catalogue.xml", and optionally an Alert Catalogue file)
    |
    |-- Pixmaps (contains XML files describing pixmaps)
    |
    |-- ColorProfiles (contains XML files with colour profiles)
    |
    |-- Symbols (contains SVG files with symbols and CSS2 style sheets)
    |
    |-- LineStyles (contains XML files with line styles)
    |
    |-- AreaFills (contains XML files area fills)
    |
    |-- Fonts (contains TrueType font files)
    |
    |-- Rules (contains files with rules which map features to drawing instructions)
```

9-13.3 Model of the Catalogue

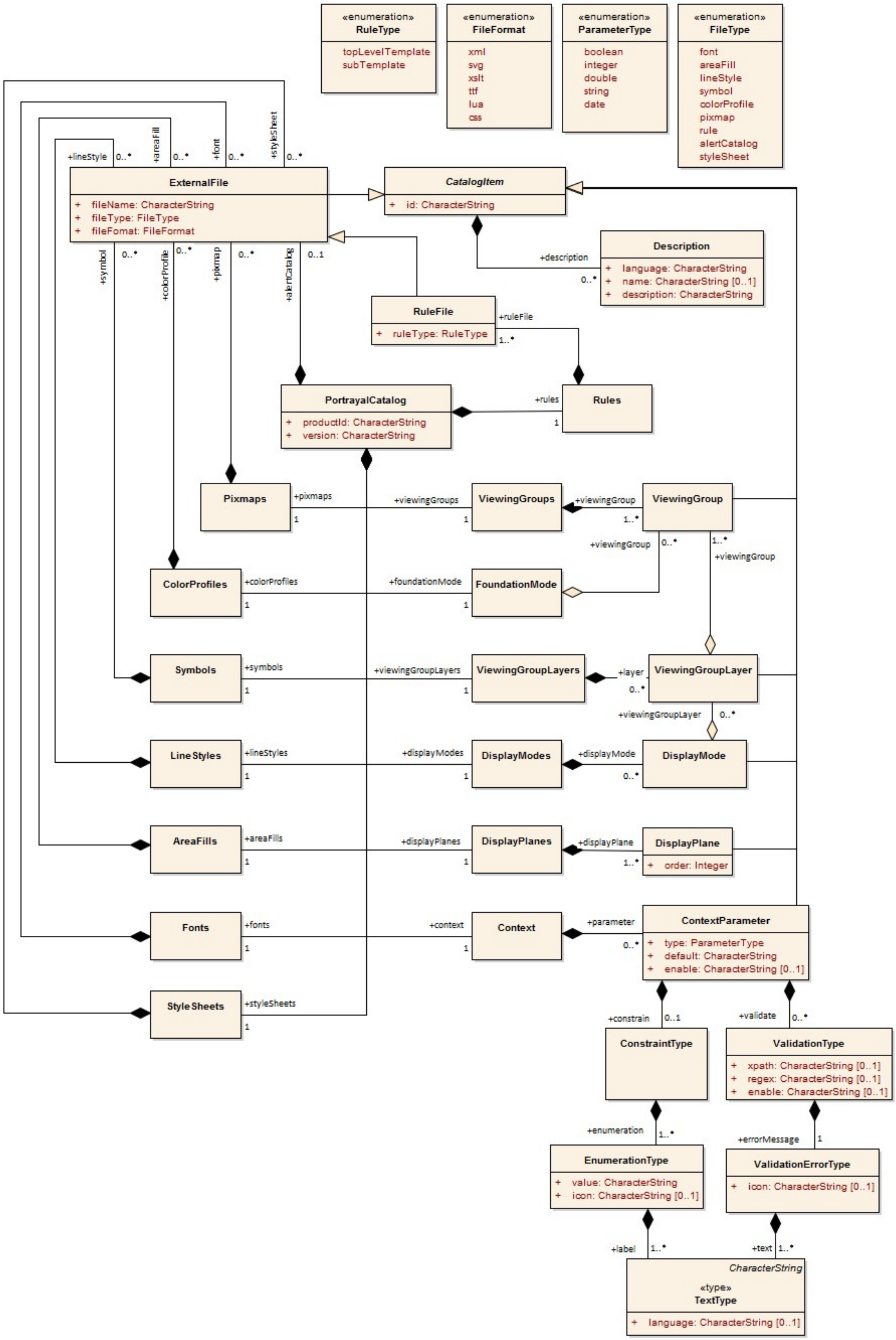


Figure 9-20 — Catalogue

9-13.3.1 PortrayalCatalog

Role Name	Name	Description	Mult.	Type
Class	PortrayalCatalog	A container of all the Catalogue items	-	-
Attribute	productId	The ID of the product for which the Catalogue is intended	1	string
Attribute	version	The version of the product the Catalogue is defined for	1	string
Role	alertCatalog	A file reference to an Alert Catalogue	0..1	ExternalFile
Role	pixmaps	Container of XML Pixmap file references	1	Pixmaps
Role	colorProfiles	Container of XML Colour Profile file references	1	ColorProfiles
Role	symbols	Container of SVG Symbol file references	1	Symbols
Role	styleSheets	Container of CSS file references	1	StyleSheets
Role	lineStyles	Container of XML Line Style file references	1	LineStyles
Role	areaFills	Container of XML Area Fill file references	1	AreaFills
Role	fonts	Container of True Type font references	1	Fonts
Role	viewingGroups	Container of viewing group definitions	1	ViewingGroups
Role	foundationMode	The definition of the foundation of the portrayal	1	FoundationMode
Role	viewingGroupLayers	Container of viewing group layers.	1	ViewingGroupLayers
Role	displayModes	Container of display mode definitions	1	DisplayModes
Role	displayPlanes	Container of display plane definitions	1	DisplayPlanes
Role	context	Container of context parameter definitions	1	Context
Role	rules	Container of rule file references	1	Rules

9-13.3.2 CatalogItem

Role Name	Name	Description	Mult.	Type
Class	CatalogItem	An abstract base class for components of the Catalogue	-	-
Attribute	id	A unique identifier of the Catalogue item	1	string
Role	description	Meta Data common to each Catalogue Item. There can be descriptions in different languages	0..*	Description

9-13.3.3 ExternalFile

Role Name	Name	Description	Mult.	Type
Class	ExternalFile	A Catalogue item that defines the reference to an external file	-	-
Subtype of	CatalogItem	See CatalogItem	-	-
Attribute	fileName	The name of the file	1	string
Attribute	fileType	The type of the file	1	FileType
Attribute	fileFormat	The format of the file	1	FileFormat

9-13.3.4 Description

Role Name	Name	Description	Mult.	Type
Class	Description	Language specific information about an item	-	-
Attribute	language	A language identifier code. ISO 639-2/T alpha-3 code (eng – English, fra – French, deu - German)	1	string

Attribute	name	An optional name of an item in the identified language	0..1	string
Attribute	description	The language specific description of the item	1	string

9-13.3.5 Pixmaps

Role Name	Name	Description	Mult.	Type
Class	Pixmaps	A container of pixmap file references	-	-
Role	pixmap	The file reference. The type is XML	0..*	ExternalFile

9-13.3.6 ColorProfiles

Role Name	Name	Description	Mult.	Type
Class	ColorProfiles	A container of colour profile file references	-	-
Role	colorProfile	The file reference. The type is XML	0..*	ExternalFile

9-13.3.7 Symbols

Role Name	Name	Description	Mult.	Type
Class	Symbols	A container of Symbol file references	-	-
Role	symbol	The file reference. The type is SVG	0..*	ExternalFile

9-13.3.8 StyleSheets

Role Name	Name	Description	Mult.	Type
Class	StyleSheets	A container of CSS file references	-	-
Role	styleSheet	The file reference	0..*	ExternalFile

9-13.3.9 LineStyles

Role Name	Name	Description	Mult.	Type
Class	LineStyles	A container of Line Style file references	-	-
Role	lineStyle	The file reference. The type is XML	0..*	ExternalFile

9-13.3.10 AreaFills

Role Name	Name	Description	Mult.	Type
Class	AreaFills	A container of Area Fill file references	-	-
Role	areaFill	The file reference. The type is XML.	0..*	ExternalFile

9-13.3.11 Fonts

Role Name	Name	Description	Mult.	Type
Class	Fonts	A container for fonts	-	-
Role	font	The file reference. For true type fonts the type is ttf.	0..*	ExternalFile

9-13.3.12 ViewingGroups

Role Name	Name	Description	Mult.	Type
Class	ViewingGroups	A container of Viewing Group definitions	-	-
Role	viewingGroup	Definition of a specific Viewing Group	1..*	ViewingGroup

9-13.3.13 ViewingGroup

Role Name	Name	Description	Mult.	Type
Class	ViewingGroup	A Viewing Group name and definition	-	-
Subtype of	CatalogItem	See CatalogItem	-	-

9-13.3.14 FoundationMode

Role Name	Name	Description	Mult.	Type
Class	FoundationMode	A set of viewing groups that forms the foundation of the portrayal and cannot be removed from the display	-	-
Role	viewingGroup	Viewing group of the foundation mode	0..*	ViewingGroup

9-13.3.15 ViewingGroupLayers

Role Name	Name	Description	Mult.	Type
Class	ViewingGroupLayers	A container of Viewing Group Layers	-	-
Role	layer	Definition of a specific Viewing Group layer	0..*	ViewingGroupLayer

9-13.3.16 ViewingGroupLayer

Role Name	Name	Description	Mult.	Type
Class	ViewingGroupLayer	A set of Viewing groups which are intended to switch on or off in an application	-	-
Subtype of	CatalogItem	See CatalogItem	-	-
Role	viewingGroup	Viewing Group of the layer	1..*	ViewingGroup

9-13.3.17 DisplayModes

Role Name	Name	Description	Mult.	Type
Class	DisplayModes	A container of Display Mode definitions	-	-
Role	displayMode	Definition of a Display Mode	0..*	DisplayMode

9-13.3.18 DisplayMode

Role Name	Name	Description	Mult.	Type
Class	DisplayMode	A set of Viewing Layers to switch on or off in an application	-	-
Subtype of	CatalogItem	See CatalogItem	-	-
Role	viewingGroupLayer	Viewing Group Layer included in this Display Mode	0..*	ViewingGroupLayer

9-13.3.19 DisplayPlanes

Role Name	Name	Description	Mult.	Type
Class	DisplayPlanes	A container of Display Plane definitions	-	-
Role	displayPlane	Definition of a Display Plane	1..*	DisplayPlane

9-13.3.20 DisplayPlane

Role Name	Name	Description	Mult.	Type
Class	DisplayPlane	A Display Plane name and definition	-	-
Subtype of	CatalogItem	See CatalogItem	-	-
Attribute	order	Used to sort the drawing order of display planes. Display planes with larger values are drawn above those with lower values Positive: Above RADAR Zero: Reserved for RADAR Negative: Below RADAR	1	integer

9-13.3.21 Context

Role Name	Name	Description	Mult.	Type
Class	Context	A container of Context Parameters	-	-
Role	parameter	Context Parameter	0..*	ContextParameter

9-13.3.22 ContextParameter

Role Name	Name	Description	Mult.	Type
Class	ContextParameter	A Context Parameter name and definition	-	-
Subtype of	CatalogItem	See CatalogItem	-	-
Attribute	type	The data type of the Parameter	1	ParameterType
Attribute	default	A default value for the Parameter	1	string
Role	constrain	Constrains the value domain of the parameter	0..1	ConstraintType
Role	validate	The validation rules for this parameter	0..*	ValidationType
Attribute	enable	XPath 1.0 Boolean expression. Used to indicate conditional parameters (for example, ShallowContour is conditional on TwoShades: “//TwoShades=false“)	0..1	string

9-13.3.23 ConstraintType

Role Name	Name	Description	Mult.	Type
Class	ConstraintType	Constrains the allowed values of a context parameter and provides meaningful names for enumerations	-	-
Role	enumeration	Valid values and labels of the constraint	1..*	EnumerationType

9-13.3.24 EnumerationType

Role Name	Name	Description	Mult.	Type
Class	EnumerationType	Represents an enumerated value	-	-
Role	label	A label for the value (for example, “Enabled“)	1..*	TextType
Attribute	value	The value of the enumeration (for example, “1“)	1	string
Attribute	icon	An icon representing the value. Reference to a catalog symbol	0..1	string

9-13.3.25 TextType

Role Name	Name	Description	Mult.	Type
Class	TextType	Represents an enumerated value	-	-
Attribute	-	String value in a national language	1	string
Attribute	language	An ISO 639-2/T alpha-3 language identifier code indicating the national language of the encoded string. Default is “eng“	0..1	string

9-13.3.26 ValidationType

Role Name	Name	Description	Mult.	Type
Class	ValidationType	Class for a validation rule. Either xpath or regex must be present, but not both	-	-
Attribute	xpath	An XPath 1.0 Boolean expression	0..1	string
Attribute	regex	A regular expression per W3C XML Standard – Part 2, Appendix F (Regular Expressions)	0..1	string
Role	errorMessage	An error message for the validation	1	ValidationErrorType
Attribute	enable	XPath 1.0 Boolean expression. Indicates conditional validation (for example, validate NationalLanguage only when populated: ‘‘//NationalLanguage[.!=]‘‘).	0..1	string

9-13.3.27 ValidationErrorType

Role Name	Name	Description	Mult.	Type
Class	ValidationErrorType	Class for a validation error	-	-
Role	text	Text of an error message in one or more national languages	1..*	TextType
Attribute	icon	An icon representing the error. Reference to a catalog symbol	0..1	string

9-13.3.28 Rules

Role Name	Name	Description	Mult.	Type
Class	Rules	A container of XSLT rule file references	-	-
Role	ruleFile	Reference to a file containing rules	1..*	RuleFile

9-13.3.29 RuleFile

Role Name	Name	Description	Mult.	Type
Class	RuleFile	Rule file reference	-	-
Subtype of	ExternalFile	See ExternalFile	-	-
Attribute	ruleType	The type of the templates within the rule file. There can be more than one top level rule which can be selected in an application to allow different portrayal of the data	1	RuleType

9-13.3.30 ParameterType

Role Name	Name	Description
Type	ParameterType	Choice of Parameter Types
Enumeration	boolean	A Boolean value
Enumeration	integer	An integer number

Enumeration	double	A floating point number
Enumeration	string	A character string
Enumeration	date	A date according to the gregorian calendar

9-13.3.31 FileFormat

Role Name	Name	Description
Type	FileFormat	The format of an external file
Enumeration	xml	
Enumeration	svg	
Enumeration	xslt	
Enumeration	ttf	
Enumeration	lua	
Enumeration	css	

9-13.3.32 FileType

Role Name	Name	Description
Type	FileType	The type of an external file
Enumeration	font	A font file
Enumeration	areaFill	A file describing an area fill
Enumeration	lineStyle	A file describing a line style
Enumeration	symbol	A file describing a symbol
Enumeration	colorProfile	A file describing a colour profile
Enumeration	pixmap	A file describing a pixmap
Enumeration	rules	A file containing portrayal rules
Enumeration	styleSheet	A file containing styles for symbols
Enumeration	alertCatalog	A file containing an Alert Catalogue

9-13.3.33 RuleType

Role Name	Name	Description
Type	RuleType	The type of templates within a rule file
Enumeration	topLevelTemplate	The rule file contains a top level template
Enumeration	subTemplate	The rule file contains templates that are used or called by other templates

9-13.4 Model of the Alert Catalogue

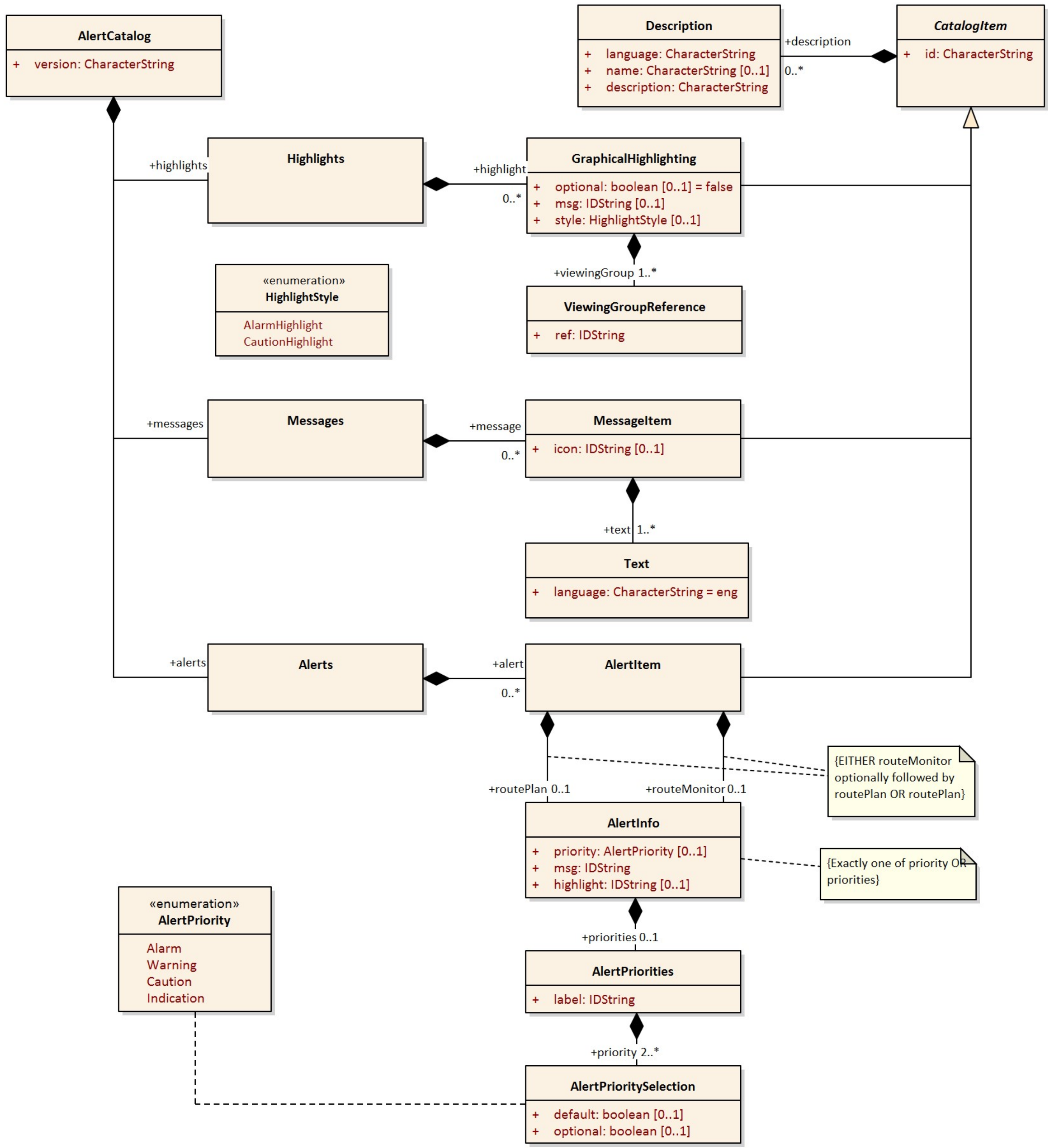


Figure 9-21 — Alert Catalogue

9-13.4.1 AlertCatalog

Role Name	Name	Description	Mult.	Type
Class	AlertCatalog	A container of all the Catalogue items	-	-
Attribute	version	The version of the Catalogue	1	string
Role	messages	Container of messages	1	Messages
Role	highlights	Container of highlights	1	Highlights
Role	alerts	Container of alerts	1	Alerts

9-13.4.2 Messages

Role Name	Name	Description	Mult.	Type
Class	Messages	A container of MessageItems	-	-
Role	message	Definition of a message	0..*	MessageItem

9-13.4.3 MessageItem

Role Name	Name	Description	Mult.	Type
Class	MessageItem	Defines a language independent message	-	-
Subtype of	CatalogItem	See CatalogItem	-	-
Attribute	icon	Reference to a symbol in the Portrayal Catalogue	0..1	string
Role	text	Language specific text	1..*	Text

9-13.4.4 Text

Role Name	Name	Description	Mult.	Type
Class	Text	Language specific string	-	-
Subtype of	string		-	-
Attribute	language	Identifies a language, default is eng. ISO 639-2/T alpha-3 code (eng – English, deu – German)	0..1	string

9-13.4.5 Highlights

Role Name	Name	Description	Mult.	Type
Class	Highlights	A container of GraphicalHighlighting items	-	-
Role	highlight	Definition of a highlight	0..*	GraphicalHighlighting

9-13.4.6 GraphicalHighlighting

Role Name	Name	Description	Mult.	Type
Class	GraphicalHighlighting	Associates display parameters with alert information	-	-
Subtype of	CatalogItem	See CatalogItem	-	-
Attribute	style	Overrides the style of graphical highlight shown in the chart area. When empty, style is inferred from the alert priority • Alarm: AlarmHighlight • Others: CautionHighlight	0..1	HighlightStyle
Attribute	optional	Allowing the highlight to be turned off is not required. Default is false	0..1	boolean
Attribute	msg	A reference to a message to be displayed while any of the viewing groups are disabled	0..1	string
Role	viewingGroup	References viewing groups used to control graphical highlighting	1..*	ViewingGroupReference

9-13.4.7 ViewingGroupReference

Role Name	Name	Description	Mult.	Type
Class	ViewingGroupReference	A reference to a viewing group	-	-
Role	ref	The identifier of the viewing group	1	string

9-13.4.8 Alerts

Role Name	Name	Description	Mult.	Type
Class	Alerts	A container of AlertItems	-	-
Role	alert	Definition of an alert	0..*	AlertItem

9-13.4.9 AlertItem

Role Name	Name	Description	Mult.	Type
Class	AlertItem	Describes a single alert	-	-
Subtype of	CatalogItem	See CatalogItem	-	-
Role	routeMonitor	The alert behavior in route monitoring	0..1	AlertInfo
Role	routePlan	The alert behavior in route planning	0..1	AlertInfo

NOTE: The content of AlertItem is implemented in the Schema as either a routeMonitor element optionally followed by a routePlan element, or a routePlan element.

9-13.4.10 AlertInfo

Role Name	Name	Description	Mult.	Type
Class	AlertInfo	The behavior of an alert in a single mode	-	-
Role	priority	A single alert priority. If present, precludes use of *priorities*	0..1	AlertPriority
Role	priorities	A set of alert priorities. If present, precludes use of *priority*	0..1	AlertPriorities

NOTE: Exactly one of priority or priorities must be present.

9-13.4.11 AlertPriorities

Role Name	Name	Description	Mult.	Type
Class	AlertPriorities	A set of alert priorities	-	-
Attribute	label	Reference to a message used to label the UI component which allows selection of the desired alert priority	1	string
Role	priority	An alert priority	2..*	AlerPrioritySelection

9-13.4.12 AlertPrioritySelection

Role Name	Name	Description	Mult.	Type
Class	AlertPrioritySelection	Adds information to an alert priority	-	-
Subtype of	AlertPriority	See AlertPriority	-	-
Attribute	default	Identifies the default priority selection. Default is false	0..1	boolean
Attribute	optional	Indicates allowing the user to choose this priority is optional. Default is false	0..1	boolean

9-13.4.13 AlertPriority

Item	Name	Description
Enumeration	AlertPriority	The priority of an alert
Value	Alarm	Indicates conditions requiring immediate attention and action by the bridge team (refer to MSC.252(83) 19.1.2)

Value	Warning	Indicates changed conditions and should be presented for precautionary reasons which are not immediately hazardous but which may become so, if no action is taken (refer to MSC.252(83) 19.1.3)
Value	Caution	Indicates a condition which does not warrant an alarm or warning condition, but still requires attention and out of the ordinary consideration of the situation or of given information (refer to MSC.252(83) 19.1.4)
Value	Indication	Display of regular information and conditions (refer to MSC.252(83) appendix 1)

9-13.4.14 HighlightStyle

Item	Name	Description
Enumeration	HighlightStyle	The style of the graphical highlight to be shown in the chart area
Value	AlarmHighlight	IEC 62288:2014 Table A.3 Navigation symbols, entry 3.5 b
Value	CautionHighlight	IEC 62288:2014 Table A.3 Navigation symbols, entry 3.5 c

9-13.5 Schema for pixmap files

A pixmap is a two dimensional array of pixels defining an image. This Schema allows to encode pixmaps that can be then be referenced, for example from pixmap area fills. The coordinate system for the pixmap is different than other coordinate systems in this standard. The y-axis is directed downwards and the origin is in the upper left corner of the pixmap.

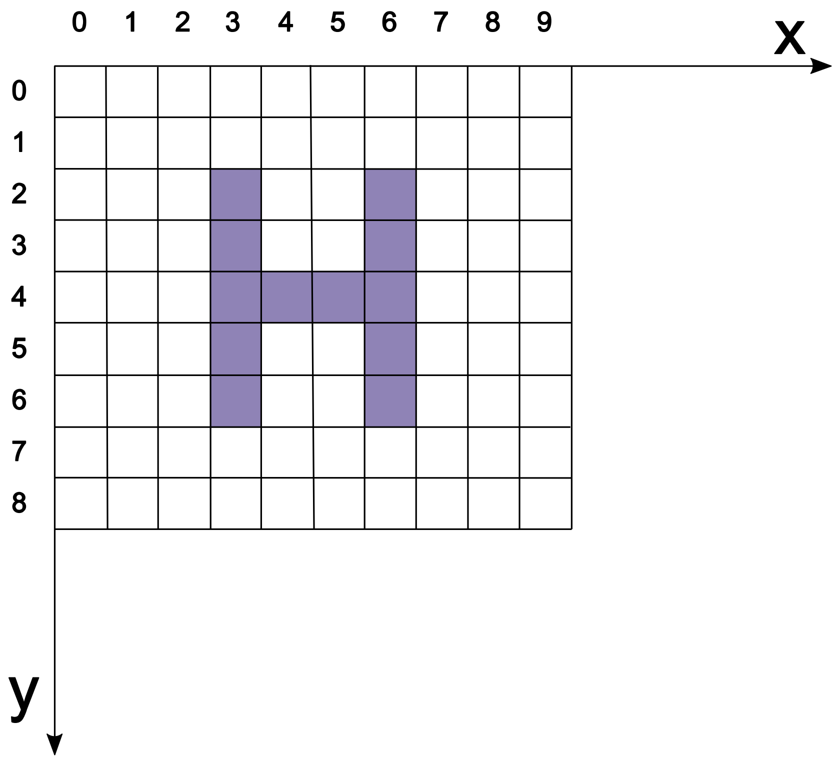


Figure 9-22 — Coordinate system for pixmap

The graphic above shows a simple pixmap with width 10 pixel and height 9 pixel. Most of the pixels are transparent (here white) some pixels are coloured.

The style defines a simple type for the colour identifier:

```
<xs:simpleType name="ColorId">
   <xs:restriction base="xs:string">
      <xs:minLength value="1"></xs:minLength>
      <xs:maxLength value="3"></xs:maxLength>
      <xs:pattern value="[a-zA-Z0-9_]+"></xs:pattern>
```

```
    </xs:restriction>
</xs:simpleType>
```

This describes a token 1 to 3 characters long that can contain digits, alpha characters or the underscore. It is used to identifiy a colour in the colour map.

The next type is a complex type for a pixel:

```
<xs:complexType name="Pixel">
    <xs:simpleContent>
        <xs:extension base="ColorId">
            <xs:attribute name="x" type="xs:nonNegativeInteger" use="required"/>
            <xs:attribute name="y" type="xs:nonNegativeInteger" use="required"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>
```

It extends the colour identifier and adds two attributes for the coordinate of the pixel according to the pixmap coordinate system.

Each pixmap contains a colour map; a list of colour definitions bundled with a colour identifier. Two types are defined in the Schema one for the colour map item and one for the colour map.

```
<xs:complexType name="ColorMapItem">
    <xs:complexContent>
        <xs:extension base="s100Symbol:Color">
            <xs:attribute name="id" type="ColorId" use="required"/>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>

<xs:complexType name="ColorMap">
    <xs:sequence>
        <xs:element name="color" type="ColorMapItem" minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
```

Note: The colour definition is taken from the S-100 symbol definition Schema. That allows using colour token from a colour profile or direct sRGB colour definitions. Transparency can be defined here as well.

The last type defined is the complex type for the pixmap itself.

```
<xs:complexType name="Pixmap">
    <xs:sequence>
        <xs:element name="description" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="width" type="xs:positiveInteger"/>
        <xs:element name="height" type="xs:positiveInteger"/>
        <xs:element name="colorMap" type="ColorMap">
            <xs:key name="colorKey">
                <xs:selector xpath="color"/>
                <xs:field xpath="@id"/>
            </xs:key>
        </xs:element>
        <xs:element name="background" type="ColorId"/>
        <xs:element name="pixel" type="Pixel" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
```

It defines an optional description element and mandatory elements for width and height. Furthermore it defines an element for the colour map, an element for the background colour and the any number of pixel elements. The background colour is implicitly used for all pixels that are not defined by a pixel element. Note that there is a key element to ensure that colour identifiers are unique.

Finally the root element is defined:

```
<xs:element name="pixmap" type="Pixmap">
  <xs:keyref refer="colorKey" name="pixelRef">
    <xs:selector xpath="pixel"/>
    <xs:field xpath="."/>
  </xs:keyref>
  <xs:keyref refer="colorKey" name="backgroundRef">
    <xs:selector xpath="background"/>
    <xs:field xpath="."/>
  </xs:keyref>
  <xs:unique name="positionUnique">
    <xs:selector xpath="pixel"/>
    <xs:field xpath="@x"/>
    <xs:field xpath="@y"/>
  </xs:unique>
</xs:element>
```

The keyref element are there for ensure the referential integrity of the colour identifier used in the pixel and background element. The unique element ensures that no pixel is defined more than ones.

A complete pixmap file for the example above looks like:

```
<?xml version="1.0" encoding="UTF-8"?>
<pixmap xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="S100Pixmap.xsd">
  <description>Test pixmap showing a capital H in faint magenta.</description>
  <width>10</width>
  <height>9</height>
  <colorMap>
    <color id="_" transparency="1.0">#000000</color>
    <color id="M">#8F83B6</color>
  </colorMap>
  <background>_</background>
  <pixel x="3" y="2">M</pixel>
  <pixel x="3" y="3">M</pixel>
  <pixel x="3" y="4">M</pixel>
  <pixel x="3" y="5">M</pixel>
  <pixel x="3" y="6">M</pixel>
  <pixel x="4" y="4">M</pixel>
  <pixel x="5" y="4">M</pixel>
  <pixel x="6" y="2">M</pixel>
  <pixel x="6" y="3">M</pixel>
  <pixel x="6" y="4">M</pixel>
  <pixel x="6" y="5">M</pixel>
  <pixel x="6" y="6">M</pixel>
</pixmap>
```

Page intentionally left blank

Appendix 9-A
Generating Portrayal Input XML
(informative)

Preface

This standard describes a base Schema that contains base types which define the expected structure of the XML input to the portrayal processing. The XSLT portrayal rules expect the input XML to conform with the base Schema, with additional information as defined within a product‘s Feature Catalogue.

Although the encoded elements of the input XML must conform to the base Schema, it is not necessary to fully describe a Schema for each product‘s input XML. It is sufficient that the portrayal rules can consistently parse the input XML using XPath 1.0 queries.

The base Schema should be used along with patterns described in this Appendix to model a generic pattern for describing all S-100 content; essentially providing a mapping from the S-100 general feature model to an XML encoding of an instance of the model.

This section describes how the input XML should be generated. Patterns will be introduced to map the data model from a Feature Catalogue to an input XML for portrayal. Although the input XML can cover the entire data model it is sufficient to generate only the part that is relevant for portrayal.

9-A-1 Input XML structure

The input XML must conform to the following structure, using the indicated element names and order.

```
<!—The root element -->
<Dataset>
        <InformationTypes>
        </InformationTypes>

        <!—Spatial Objects -->
        <Points>
        </Points>
        <MultiPoints>
        </MultiPoints>
        <Curves>
        </Curves>
        <CompositeCurves>
        </CompositeCurves>
        <Surfaces>
        </Surfaces>

        <Features>
        </Features>
</Dataset>
```

9-A-2 Simple attributes

When encoding simple attributes the element name should match the code of a simple attribute defined within the product‘s Feature Catalogue. For enumerations, the code should be passed as the value, the label should not be used.

```
<colour>3</colour> <!—don’t encode as red -->
```

9-A-3 Complex attributes

When encoding complex attributes the element name should match the code of a complex attribute defined within the product's Feature Catalogue. Simple and/or complex sub attributes should be nested within the element.

An instance of complex attribute “speed“ with simple sub attributes “speedMaximum“ and “speedMinimum“:

```
<speed>
  <speedMaximum>10</speedMaximum>
  <speedMinimum>0</ speedMinimum>
</speed>
```

9-A-4 Information types

Instances of information types are encoded within the <*InformationTypes*> section of the input XML. Information types are referenced by other objects encoded in the input XML.

The base Schema defines abstract type *Information* for encoding of information types. The following pattern describes instantiation of the abstract type:

- The element name of each information type object must match the code of an InformationType defined in the product's Feature Catalogue. The code defines the instantiation of the abstract type; in the example below, ChartNote is the instantiation of *Information*.
- The value(s) of the attributes of the information type should be included in the encoding, as described in clauses 9-A-2 and 9-A-3. These values should be nested within the element described by the preceding bullet (they are part of the instantiation).

Notional information type ChartNote, described in the Feature Catalogue as having complex attribute *note*, which has simple sub attributes *noteText* and *language*:

```
<ChartNote id="I1">
  <note>
    <noteText>Hello world!</noteText>
    <language>en</language>
  </note>
  <note>
    <noteText>Hallo Welt!</noteText>
    <language>de</language>
  </note>
</ChartNote>
```

9-A-5 Spatial objects

The base Schema defines spatial types which describe XML encodings of the Part 7 Spatial Schema. These types provide a consistent way to represent spatial objects to the portrayal processing. The input XML encoding of spatial objects should conform with the base Schema; the spatial model cannot be modified by a product's Feature Catalogue.

In most cases the portrayal rules do not attempt to reference the geometry coordinates of spatial objects; in these cases the geometry coordinates may be omitted from the input XML, potentially providing significant advantages to the speed of generating and parsing the input XML. A notable exception when the geometry coordinates cannot be omitted is symbolization of MultiPoint spatial objects, as geometry coordinates should always be provided for MultiPoint spatial objects.

9-A-6 Features

Instances of feature types are encoded within the <*Features*> section of the input XML.

The base Schema defines abstract type *Feature* for encoding of feature objects. The following pattern describes instantiation of the abstract type:

- The element name of each feature object must match the code of a feature type defined in the product's Feature Catalogue. The code defines the instantiation of the abstract type; in the example below, BeaconLateral is the instantiation of *Feature*.
- The value(s) of the attributes of the feature object should normally be included in the encoding, as described in clauses 9-A-2 and 9-A-3. These values should be nested within the element described by the preceding bullet (they are part of the instantiation). Attribute values provided via Coverages should be omitted from the encoding.
- All references to spatial objects should be included as described in the base Schema.
- All associations to information type instances and feature instances should be included, as described in clause 9-A-7.

An example encoding of an instance of a BeaconLateral feature. The feature references a spatial object of type Point (P23), and has an association to a feature instance (F8) which should also be described in the input XML, but is not shown in this example. The value of each of the features attributes is also encoded.

```
<BeaconLateral id="F4" primitive=”Point” >
  <Point ref="P23" scaleMinimum=“4294967295” scaleMaximum=”0” />
  <beaconShape>1</beaconShape>
  <categoryOfLateralMark>2</categoryOfLateralMark>
  <colour>3</colour>
  <reportedDate>20050124</reportedDate>
  <status>1</status>
  <scaleMinimum>89999</scaleMinimum>
  <StructureEquipment role=”supports” featureRef=”F8” />
</BeaconLateral>
```

9-A-7 Associations

Associations are named relationships between objects. There are two types of associations: information associations for relationships between any object and an information type, and feature associations for relationships between two feature types.

The base Schema provides element *associatedInformation* as part of any object, and provides element *associatedFeature* as part of feature objects. The pattern for encoding associations is:

- The element name of each association must match the code of an association defined in the product's Feature Catalogue. The code describes the subtype of the association; in the example below, StructureEquipment replaces *associatedFeature* from the base Schema and describes the relationship between the two feature objects.
- The value of the *role* attribute should match a role described in the Feature Catalogue.
- The value of *featureRef* (for feature associations) or *informationRef* (for information associations) should match the unique id of a feature instance or information type instance encoded within the input XML.
- Include the values of any simple or complex attributes which are defined for the association as described in the Feature Catalogue. These values should be nested within the association element.

An example feature association: *StructureEquipment*. The association should be nested within the encoding of a feature instance.

```
<Features>
  <FeatureTypeX id="F1" primitive=”Point” >
    <!-- feature attributes and spatial reference omitted -->
    <StructureEquipment role=”supports” featureRef=”F2”/>
  </FeatureTypeX>
  <FeatureTypeY id="F2" primitive=”Point” >
    <!-- feature attributes and spatial reference omitted -->
    <StructureEquipment role=”supportedBy” featureRef=”F1” />
  </FeatureTypeY>
</Features>
```

An example information association: An instance of a *SpatialQuality* information type referenced from a spatial object using a *SpatialAssociation* InformationAssociation type:

```xml
<InformationTypes>
  <SpatialQuality id="I1">
    <qualityOfHorizontalMeasurement>4</qualityOfHorizontalMeasurement>
  </SpatialQuality>
</InformationTypes>
<Points>
  <Point id="P1">
    <SpatialAssociation role=”defines” informationRef=”I1” />
    <Coordinate2D>
      <x>0.0</x>
      <y>1.0</y>
    </Coordinate2D>
  </Point>
</Points>
```

Appendix 9-B **SVG Profile** (normative)

9-B-1 Introduction

This appendix describes the subset of SVG elements that have been used in the creation of S-100 SVG symbols and covers the set of SVG elements and associated attributes and properties that are in use by S-100.

The S-100 SVG profile is a subset of the SVG Tiny 1.2 profile
http://www.w3.org/TR/SVGTiny12/

9-B-2 Top Level SVG

The main svg element carries this indication as well as properties of each individual svg symbol as xml attributes.

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.2" baseProfile="tiny" xml:space="preserve"
style="shape-rendering:geometricPrecision; fill-rule:evenodd;" width="4.34mm" height="5.35mm"
viewBox="-2.22 -2.79 4.34 5.35">
```

9-B-2-1 Coordinate System

The overall width and height of the symbol are defined in mm. The viewbox covers the range of coordinates used for the symbol. The pivot point of the symbol is designed to be at the 0,0 position.

The default coordinate system used for S-100 SVG has the origin in the upper left corner with the x-axis pointing to the right and the y-axis pointing down.

9-B-2-2 Title

The title element is used to carry the name of the symbol.

```
<title>ACHARE02</title>
```

9-B-2-3 Description

The description element is used to carry a brief textual description of the symbol.

```
<desc>anchorage area as a point at small scale, or anchor points of mooring trot at large
scale</desc>
```

9-B-2-4 Metadata

SVG has a metadata element which allows for the direct inclusion of metadata document fragments from other namespaces. The following example shows how IHO could define the appropriate metadata content for a symbol. The IHO S-100 working group is encouraged to define a metadata Schema for use with S-100 symbols.

```
 <metadata>
  <iho:S100SVG xmlns:iho="http://www.iho.int/SVGMetadata">
   <iho:Description iho:publisher="IHO" iho:creationDate="2014-06-09" iho:source="S52Preslib4.0"
iho:format="S100SVG" iho:version="0.1"/>
  </iho:S100SVG>
 </metadata>
```

9-B-3 Drawing Elements

The body of the SVG symbol contains the drawing elements. The drawing elements implemented so far include path, rect and circle with details to follow. These drawing elements share some common attributes such as ‘class’.

9-B-3-1 Class

The ‘class’ attribute is used to assign one or more class names to the element. In the S-100 SVG the class attribute is used to assign style information by way of a CSS stylesheet. It can also be used to filter or control which elements should be shown. Essentially the class tokens can be used as a key to find a set of style instructions in the corresponding Cascading Style Sheet (CSS). A processing instruction at the head of the SVG symbol indicates the corresponding CSS file.

9-B-3-1.1 CSS

```
<?xml-stylesheet href="SVGStyle.css" type="text/css"?>
```

An example excerpt of such a CSS file might be as follows:

```
.layout {display:none} /* used to control visibility of symbolBox, svgBox, pivotPoint (none or inline) */
.symbolBox {stroke:black;stroke-width:0.32;} /* show the cover of the symbol graphics */
.svgBox {stroke:blue;stroke-width:0.32;} /* show the entire SVG cover */
.pivotPoint {stroke:red;stroke-width:0.64;} /* show the pivot/anchor point, 0,0 */
.sl {stroke-linecap:round;stroke-linejoin:round} /* default line style elements */
.f0 {fill:none} /* no fill */
.sCURSR {stroke:#E38039} /* sRGB line colour for colour token CURSR */
.fCURSR{fill:#E38039} /* sRGB fill colour for colour token CURSR*/
.sCHBLK {stroke:#000000}
.fCHBLK {fill:#000000}
.sCHGRD {stroke:#4C5B63}
.fCHGRD {fill:#4C5B63}
.sCHGRF {stroke:#768C97}
.fCHGRF {fill:#768C97}
.sCHRED {stroke:#EA5471}
.fCHRED {fill:#EA5471}
.sCHGRN {stroke:#52E93A}
.fCHGRN {fill:#52E93A}
.sCHMGD {stroke:#C045D1}
.fCHMGD {fill:#C045D1}
…
```

This mechanism allows for possibility to change the colours used in the symbols by swapping the CSS file with different contents according to the desired colour scheme. Each colour token is encoded for both a stroke style and a fill style. The stroke is used for drawing lines and the fill for filling closed shapes. In the above example the token ‘sCHMGD’ translates into a ‘stroke’ property using the sRGB colour #C045D1 and fCHMGD represents a fill operation. Different CSS files would be used for each colour palette with the sRGB values calculated using a formula to convert from the official CIE values.

NOTE: In converting CIE to sRGB, the rendering intent must follow an absolute colorimetry method. Due to the differences in colour and luminance performance between individual monitors, any “formula“ for conversion from CIE to sRGB must be based on measurements to characterize (calibrate) the monitor in order to meet the colour accuracy and separation specified for ECDIS. For interoperability with ECDIS, portrayal of other S-1xx products would need to follow the same rendering intent.

9-B-3-2 Style Properties

The style properties used in the draft S-100 SVG symbols include:

- ‘stroke’ - the pen colour for lines defined with a hexadecimal sRGB value;
- ‘stroke-width’ - the pen width in the same units as the SVG width/height. For S-100 SVG we are using mm;
- ‘stroke-opacity’ - range of 0.0 (fully transparent) to 1.0 (fully opaque);
- ‘fill’ - the colour to fill closed shapes defined with a hexadecimal sRGB value;
- ‘fill-opacity’- range of 0.0 (fully transparent) to 1.0 (fully opaque);

- ‘stroke-linecap’ – style for the ends of lines, choice of (butt | round | square);
- ‘stroke-linejoin’ – style for corners within a line path, choice of (miter | round | bevel);
- ‘display’ – identifies whether the element is to be included/rendered or not. Default is ‘inline’. This is used as a way to hide or show the layout elements of the symbol such as the covering box or the pivot point. This way a different CSS file with layout display set to ‘inline’ can be used when viewing the symbol in a design/engineering view.

9-B-3-3 Path

<path d=" M -2.06,1.36 L -1,2.4 L 0.98,2.4 L 1.96,1.39" class="sl f0 sCHMGD" style="stroke-width: 0.32;"/>

<path d=" M -5.88,-5.88 L 5.87,-5.88 L 5.87,5.87 L -5.88,5.87 L -5.88,-5.88 Z" class="fDNGHL" style="fill-opacity:0.25;"/>

The ‘d’ attribute carries the path data which describes the outline of a shape. In the current set of SVG symbols the path data is made up of *moveto* ‘M’ and *lineto* ‘L’ instructions as well as the *closepath* ‘Z’ instruction. The *curve* instructions have not yet been used. ‘M’ and ‘L’ instructions are following by a pair of absolute coordinates. Relative coordinates indicated with lowercase ‘m’ and ‘l’ instructions are not used. Note that some style elements can be assigned specifically using the ‘style’ attribute and others are coming from the stylesheet via the class lookups as described above.

9-B-3-4 Rectangle

<rect class="symbolBox layout" fill="none" x="-2.06" y="-2.63" height="5.03" width="4.02"/>

The ‘rect’ command uses ‘x’ and ‘y’ to define the upper left corner of the rectangle and the attributes ‘width’ and ‘height’ in the user units, mm. Specific style parameters are defined using the ‘style’ attribute while colours and other common styles are applied via the class token CSS lookups.

9-B-3-5 Circle

<circle class="pivotPoint layout" fill="none" cx="0" cy="0" r="1"/>

The ‘circle’ command uses ‘cx’ and ‘cy’ to define the centre of the circle and the attribute ‘r’ to define the radius in the user units, mm. Specific style parameters are defined using style attributes while colours and other common styles are applied via the class token CSS lookups.

Page intentionally left blank

S-100 – Part 9a

Portrayal (Lua)

Page intentionally left blank

Contents

Page intentionally left blank

9a-1 Scope

This part defines the changes to S-100 Part 9 necessary to implement portrayal using the scripting mechanism defined in S-100 Part 13. Products which specify use of a portrayal catalogue as described in this part must also require implementation of S-100 Part 13.

9a-2 Conformance

This part of the specification conforms to S-100 Part 13.

9a-3 Normative references

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

Lua 5.1 Reference Manual, https://www.lua.org/manual/5.1/

9a-4 Portrayal Catalogue

There are no changes to the Part 9 portrayal catalogue overview.

9a-5 General Portrayal Model

There are no changes to the Part 9 general portrayal model. A Lua portrayal follows the general portrayal model described in 9-5. Figure 9a-1 illustrates the general portrayal model.

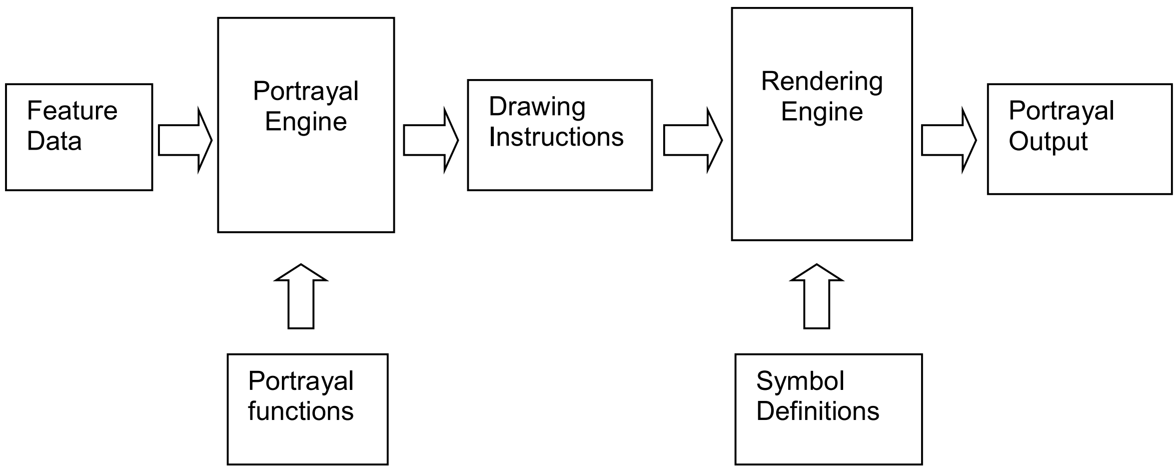


Figure 9a-1 – General portrayal model

9a-5.1 The Portrayal Process

As illustrated in Figure 9a-2, a Lua portrayal requires the following changes to the portrayal process described in Part 9, clause 9-5.1 and captured in Table 9a-1:

Table 9a-1 - Changes to the portrayal process

Part 9	Part 9a
Portrayal functions are written in the XSLT programming language.	Portrayal functions are written in the Lua programming language.
Host provides an XSLT implementation.	Host provides a Lua interpreter or Lua virtual machine.
Feature data is exposed to the portrayal functions via an XML document which must describe all features to be portrayed, along with all attribution, spatial relations, information associations, and all other information which may be used by the portrayal functions.	Feature data is not initially exposed to the portrayal functions. Instead, the host provides a list of the feature IDs to be portrayed; the portrayal functions will request attribution, spatial relations, information associations, and all other information as needed via host call-back functions.
Drawing instructions are returned to the host as an XML document, which is the result of an XSL transformation applied to the input feature data.	Drawing instructions are returned to the host via host call-back function *HostPortrayalEmit*.

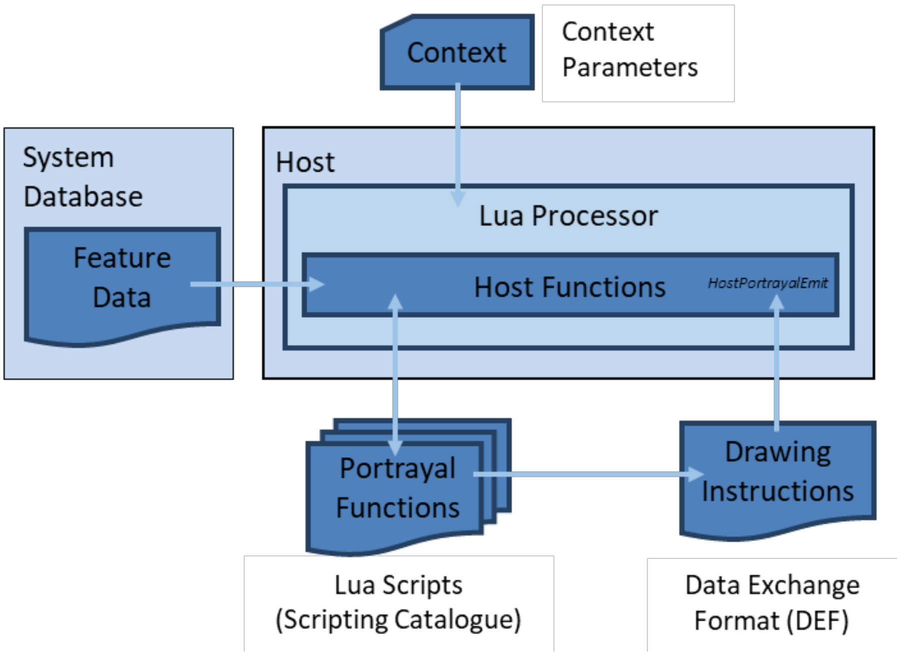


Figure 9a-2 – Portrayal process

9a-5.2 Lua Portrayal Process

This section describes the Part 9a portrayal process in detail, and indicates where there are changes to Part 9. The Lua portrayal process is shown in Figure 9a-3 .

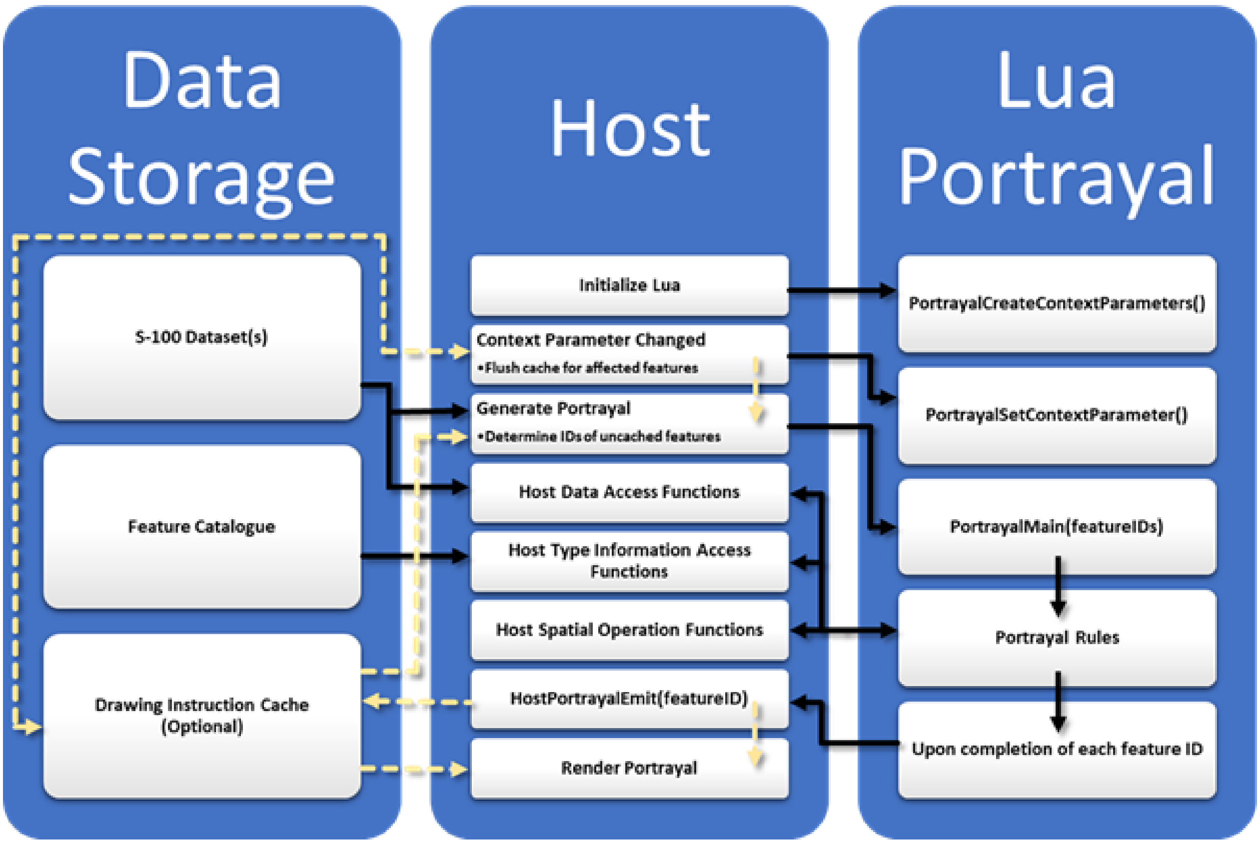


Figure 9a-3 - Lua Portrayal Process

9a-5.2.1 Portrayal Initialization

Prior to calling Lua portrayal functions, the host must register the domain specific scripting catalogue functions by loading a portrayal catalogue *TopLevelTemplate* rule file (a Lua script file). In order to prevent name collisions on *PortrayalMain*, the host must instantiate and initialize a new Lua runtime environment each time the *TopLevelTemplate* is changed. Alternatively, the host can maintain multiple Lua runtimes, one for each *TopLevelTemplate*.

After registering the scripting catalogue functions, the host calls *PortrayalInitializeContextParameters*, passing in the name and default value for each portrayal context parameter defined by the portrayal catalogue. The portrayal context parameter values are associated with the given dataset and stay in effect until the scripting session is closed, or the values are changed via *PortrayalSetContextParameter*.

9a-5.2.2 Generating a Portrayal

Portrayal script function *PortrayalMain* (see clause 9a-14.1.1) is used to generate drawing instructions for a set of feature instances. The host passes in a set of feature IDs to *PortrayalMain*; the portrayal scripts will iterate over the feature IDs and generate drawing instructions for each.

As each feature instance is processed, the portrayal engine will call standard host functions to request attribute, spatial, or other information as needed. Upon completion of processing for a feature instance the portrayal engine will call *HostPortrayalEmit* (see clause 9a-14.2.1) and provide the drawing instructions for that feature instance to the host application.

The portrayal for a given *S100_Dataset* is complete when the call to *PortrayalMain* returns. If the portrayal completed successfully, *PortrayalMain* returns true, otherwise *PortrayalMain* returns false along with a message indicating why the portrayal did not run to completion.

A host can terminate a portrayal prior to processing all feature instances by returning false from *HostPortrayalEmit*.

Calling *PortrayalMain* with all feature IDs from a given dataset will generate drawing instructions for the entire dataset. Drawing instructions for a subset of a dataset can be (re)generated by passing in feature IDs corresponding to the subset. This is useful when the host needs to regenerate a set of cached drawing instructions, or if the host is portraying a subset of a dataset such as a single *S100_DataCoverage*.

9a-5.2.2.1 Implementing a Portrayal Cache

In order to speed up the rendering process the host can optionally implement a portrayal cache. A portrayal cache is used to cache the drawing instructions which are output from the portrayal. Caching the drawing instructions for each feature instance allows the host to re-render feature instances without re-generating their portrayal. A cached drawing instruction only needs to be re-generated when one or more context parameters which were used to generate the drawing instruction changes.

When the portrayal scripts return the drawing instructions for a feature instance they also return a list of “observed” portrayal context parameters (see clause 9a-14.2.1). The observed context parameters are those context parameters which were evaluated during the generation of drawing instructions for a particular feature. For more detail on context parameters refer to Part 9 clause 9-13.3.22.

A notional portrayal cache is shown in Figure 9a-4. To implement, the host should cache the value of observed context parameters along with the generated drawing instructions and associate both with the feature instance. Note that a feature instance may have any number of observed context parameters, including zero.

Any changes to a context parameter requires that the host regenerate the drawing instructions for all feature instances with a matching observed context parameter. Alternatively, the host may use cached drawing instructions which were previously generated for the new value of the changed context parameter(s). Features which have no observed parameters can persist in the cache until a new portrayal catalogue is issued.

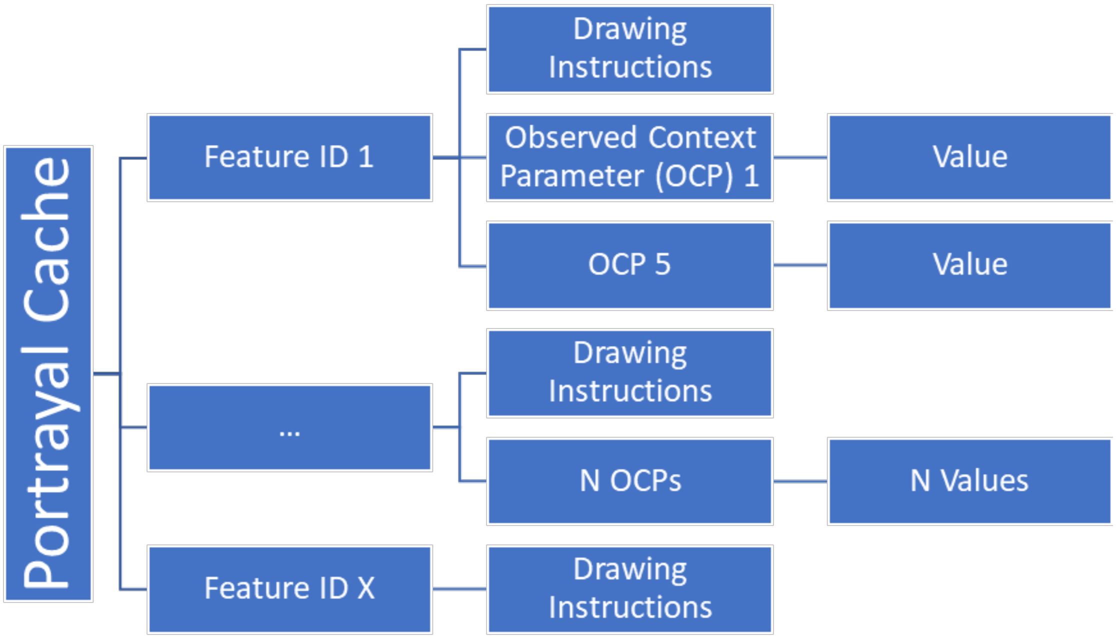


Figure 9a-4 - Notional Portrayal Cache

9a-5.2.2.2 Pre-processing a Portrayal

Implementing a portrayal cache allows the host to pre-generate the drawing instructions for a given set or sets of context parameters. This would typically be implemented as part of the hosts data import functionality.

9a-6 Package Overview

There is no change to the Part 9 package overview, although most packages are unused by Part 9a due to the removal of the portrayal input schema.

9a-7 Data input schema

This part does not use a data input schema as defined in Part 9 clause 9-7. Data is passed between a 9a portrayal and a host as described in Part 13.

9a-8 Information objects

Information objects as described in Part 9 are unused in Part 9a. Instead, information associated with features to be portrayed is obtained as described in Part 13.

9a-9 Feature objects

Feature objects as described in Part 9 are unused in Part 9a. Instead, all features are retrieved from the host as described in Part 13.

9a-10 Portrayal processing

The XSLT processing described in Part 9 clause 9-10 is replaced with Lua as described in Part 13.

9a-11 Drawing Instructions

Drawing instructions are provided to the host using DEF as described in Part 13 clause 13-6.1. A single drawing instruction is equivalent to a single DEF element.

This section describes the model and schema for drawing instructions.

9a-11.1 The concepts of drawing instructions

9a-11.1.1 General concept

As in Part 9, the output of the portrayal engine is a set of drawing instructions. These typically link the feature instance to a symbol and/or alert reference. The geometry is either taken from the feature type or can be generated by the portrayal functions. The latter is supported by the concept of augmented geometry as described in Part 9 clause 9-11.1.13 Augmented Geometry.

The conceptual model for Part 9a drawing instructions is a command-driven state machine. This model is consistent with both SVG and S-52 DAI, but differs from Part 9 which uses stateless drawing instructions.

To implement Part 9a drawing instructions, the host must maintain state while executing the drawing instructions for a given feature instance. For example, if a drawing instruction sets a pen colour, that pen colour should also be used for subsequent draw instructions. The state must be reset prior to executing the drawing instructions for each feature instance.

9a-11.2 Model of the Drawing Instructions

As in Part 9, this section describes the output of the portrayal functions. A single domain-specific scripting host function, see clause 9a-14.2.1 *HostPortrayalEmit*, provides the drawing instructions for each feature instance.

Each drawing instruction is encoded in a DEF element as described in Part 13, clause 13-6.1. A drawing instruction is an ordered pair comprised of a command and a parameter list. The command is encoded in a DEF item, and the commands parameters are encoded in a DEF parameter list.

Table 9a-2 – DEF encoding of Drawing Instructions

Portrayal Item	DEF Encoding	Example
Drawing Instruction	Element	FillColor:CHBRN,0
Command	Item	FillColor
Parameter List	Parameter List	CHBRN,0
Parameter	Parameter	CHBRN

Each drawing instruction contains a single case sensitive command. Each command has zero or more parameters.

There are two types of commands: drawing commands and state commands. Drawing commands instruct the host to render graphics. State commands instruct the host to set the state for subsequent drawing commands.

Each command and its parameters are described in the following sub-sections, grouped by purpose. In the tables which follow, the Type column is as described in Part 13 table 13-7. The **X-Ref** column refers to the equivalent Part 9 drawing instruction concept. The Part 9 reference may contain relevant information such as range of expected values or units.

9a-11.2.1 Drawing Commands

Drawing commands are used to render graphics. They are analogous to realizations of the Part 9 clause 9-11.2 *DrawingInstruction* class. The drawing commands are listed in Table 9a-3 and each command is described on the following pages.

Table 9a-3 – Drawing Commands

Command	Parameters	Parameter Type	Part 9 Reference
PointInstruction	symbol	string	9-11.2.7 9-11.2.13
LineInstruction	lineStyle	string	9-11.2.8 9-11.2.15 9-11.2.16
LineInstructionUnsuppressed	lineStyle	string	9-11.2.8 9-11.2.15 9-11.2.16
ColorFill	Token	string	9-11.2.9 9-12.5.1.4 9-11.2.17
	transparency	double	
AreaFillReference	reference	string	9-11.2.9 9-12.5.1.3 9-11.2.17
PixmapFill	reference	string	9-11.2.9 9-12.5.1.5 9-11.2.17
SymbolFill	symbol	string	9.11.2.9 9-12.5.1.6 9-11.2.17
	v1	Vector	
	v2	Vector	
	clipSymbols	boolean	
HatchFill	direction	Vector	9.11.2.9 9-12.5.1.7 9-11.2.17
	distance	double	
	lineStyle	string	
TextInstruction	text	string	9-11.2.10 9-11.2.12
CoverageFill	attributeCode	string	9-11.1.12 9-11.2.11 9-12.7.4.7
	uom	string	
	placement	string	
NullInstruction	-	-	9-11.2.6

The graphic rendering of each drawing command can be modified by preceding state commands, as described in clause 9a-11.2.2.

PointInstruction:*symbol*

Instructs the host to draw a Portrayal Catalogue symbol, placed as follows:

Table 9a-4 – PointInstruction Symbol Placement

Geometry Type	Symbol Placement
Point	At the point, then apply *LocalOffset*
Line	Along the line by *LinePlacement*, then apply *LocalOffset*

Area	At *AreaCRS*, then apply *LocalOffset*. Note that this can cause the symbol to be drawn at multiple locations

LineInstruction:*lineStyle[,lineStyle,…]*

Instructs the host to stroke a line or area geometry using the specified linestyle(s).

The host must ensure line segments with lower drawing priority are suppressed (not drawn) when coincident line segments with higher drawing priority are drawn.

Each linestyle parameter refers to either a linestyle defined within the Portrayal Catalogue or to a linestyle created by a preceding *LineStyle* command.

Note: Part 10 clause 10a-5.10.1 defines how masked spatial elements are encoded in a dataset. When executing this instruction the host must suppress the portrayal of masked spatial elements.

LineInstructionUnsuppressed:*lineStyle[,lineStyle,…]*

Instructs the host to stroke a line or area geometry using the specified linestyle(s).

The line segments should be drawn without regard for coincident line segments.

Each linestyle parameter refers to either a linestyle defined within the Portrayal Catalogue or to a linestyle created by a preceding *LineStyle* command.

Note: Part 10 clause 10a-5.10.1 defines how masked spatial elements are encoded in a dataset. When executing this instruction the host must suppress the portrayal of masked spatial elements.

ColorFill:*token[,transparency]*

Instructs the host to fill an area using the given colour token and transparency. If transparency is not given, a value of zero is assumed.

AreaFillReference:*reference*

Instructs the host to fill an area using *areaFill* (Part 9 clause 9-13.3.10) defined within the Portrayal Catalogue.

PixmapFill:*reference*

Instructs the host to fill an area using *pixmap* (Part 9 clause 9-13.3.5) defined within the Portrayal Catalogue.

A preceding *AreaCRS* command may set the origin of the pattern.

SymbolFill:*symbol,v1,v2[,clipSymbols]*

Instructs the host to fill an area using a symbol defined within the Portrayal Catalogue. A preceding *AreaCRS* command may set the origin of the pattern.

symbol The symbol used for the pattern.

v1 The offset of the next symbol in the first dimension of the pattern according to the local CRS.

v2 The offset of the next symbol in the second dimension of the pattern according to the local CRS.

clipSymbols If true (or missing), symbols are clipped at area boundaries. If false, symbols extending over the area boundaries are not drawn at all.

HatchFill:*direction,distance,lineStyle[,lineStyle]*

Instructs the host to fill an area using a hatch symbol defined within the Portrayal Catalogue. Direction and distance are as defined in Part 9 clause 9-12.5.1.8.

Each linestyle parameter refers to either a linestyle defined within the Portrayal Catalogue or to a linestyle created by a preceding *LineStyle* command.

A preceding *AreaCRS* command may set the origin of the pattern.

direction The vector defining the direction of the set of lines.

distance The distance between the lines measure perpendicular to the direction.

lineStyle A reference to a line style used for each hatch line.

TextInstruction:*text*

Instructs the host to draw the specified text placed as follows:

Table 9a-5 – TextInstruction Initial Placement

Geometry Type	Initial Placement
Point	Relative to the point
Line	Relative to the line as determined by *LinePlacement*
Area	Relative to *AreaCRS*. Note that this can cause the text to be drawn at multiple locations

Once the initial positioning is determined, the text is offset as specified by state commands *LocalOffset* and *TextVerticalOffset*. The text is aligned as specified by state commands *TextAlignHorizontal* and *TextAlignVertical*.

If preceded by a *FontReference* command the font is as specified in the Portrayal Catalogue. Otherwise the host should construct a font using the values specified by preceding *FontColor, FontSize, FontProportion, FontWeight, FontSlant, FontSerifs* and *FontStrikethrough* state commands.

text The text to display.

CoverageFill:*attributeCode[,uom[,placement]]*

Instructs the host to fill a coverage using the lookup table entries created via the *LookupEntry* state command. The host must clear the coverage lookup list upon completion.

attributeCode Specifies which of the features attributes to use for the lookup.

uom If present, specifies the unit of measure for the range values in the lookup table. If not present, the range values and attribute value share the same unit of measure as defined in the Feature Catalogue.

placement If present, specifies the placement of symbol/text annotations.

NOTE: When associating alerts with coverage values there may or may not be portrayal elements present in the coverage lookup list.

NullInstruction

Used to indicate a feature is purposefully not portrayed. Can associate an alert with a feature or geometry which is not portrayed, or whose alert geometry is different than the portrayed geometry.

9a-11.2.2 State Commands

State commands are used to set or modify the state for drawing commands which follow. To implement the portrayal the host should associate each parameter of a state command with a variable; each state command modifies the value of one or more of these variables.

The host should set the initial state as indicated in the tables of the following subsections. The state should be reset prior to executing the drawing instructions for each feature instance.

For each state command listed in the following sub-sections the applicability is given; this indicates which commands use the variables set by the state command.

Table 9a-6 shows the different types of state commands.

Table 9a-6 – Types of State Commands

Command Type	Command	Purpose
Visibility	ViewingGroup	Modifies the visibility and drawing order of drawing commands
	DisplayPlane	
	DrawingPriority	
	ScaleMinimum	
	ScaleMaximum	
	Id	
	Parent	
	Hover	
Transform	LocalOffset	Applies transformations to elements drawn by drawing commands
	LinePlacement	
	AreaPlacement	
	AreaCRS	
	Rotation	
	ScaleFactor	
Line Style	LineStyle	Defines linestyles for use by drawing commands
	LineSymbol	
	Dash	
Text Style	FontColor	Modifies the appearance of text drawn by drawing commands
	FontSize	
	FontProportion	
	FontWeight	
	FontSlant	
	FontSerifs	
	FontUnderline	
	FontStrikethrough	
	FontUpperline	
	FontReference	
	TextAlignHorizontal	
	TextAlignVertical	
	TextVerticalOffset	
Colour Override	OverrideColor	Overrides the colours defined within a symbol or pixmap referenced by drawing commands
	OverrideAll	
Geometry	SpatialReference	Defines new geometries (augmented geometry) or restricts the geometry used by drawing commands
	AugmentedPoint	
	AugmentedRay	

	AugmentedPath	
	Polyline	
	Arc3Points	
	ArcByRadius	
	Annulus	
	ClearGeometry	
Coverage	LookupEntry	Defines lookup entries which can be referenced by the *CoverageFill* drawing command
	NumericAnnotation	
	SymbolAnnotation	
	CoverageColor	
Time	Date	Defines time intervals
	Time	
	DateTime	
	TimeValid	
	ClearTime	
Alert	AlertReference	Associates geometries with alert catalogue entries

9a-11.2.2.1 Visibility Commands

Visibility commands affect the visibility and drawing order of all subsequent drawing commands. They correspond to attributes of the Part 9 clause 9-11.2.2 *DrawingInstruction* class.

Table 9a-7 – Visibility Commands

Command	Parameters	Type	Initial State	Part 9	Notes
ViewingGroup	viewingGroup	string[]	""-	9-11.1.3	For example: 21000
	…				
	viewingGroupN				
DisplayPlane	displayPlane	string	""	9-11.1.5	For example: overRadar
DrawingPriority	drawingPriority	integer	0	9-11.1.6	
ScaleMinimum	scaleMinimum	integer	max integer	9-11.2.2	
ScaleMaximum	scaleMaximum	integer	min integer	9-11.2.2	
Id	id	string	“”	9-11.2.2	
Parent	id	string	“”	9-11.2.2	
Hover	hover	boolean	false	9-11.2.2	

ViewingGroup:*viewingGroup[,viewingGroup2[,…]]*

Sets the viewing group(s) for drawing commands which follow. The drawing command is disabled if any viewing group is disabled.

Applicability: All drawing commands

DisplayPlane:*displayPlane*

Sets the display plane for drawing commands which follow.

Applicability: All drawing commands except *NullInstruction*

DrawingPriority:*drawingPriority*

Sets the drawing priority for drawing commands which follow.

Applicability: All drawing commands except *NullInstruction*

ScaleMinimum:*scaleMinimum*

Sets the scale denominator defining the minimum scale for drawing commands which follow.

Applicability: All drawing commands except *NullInstruction*

ScaleMaximum:*scaleMaximum*

Sets the scale denominator defining the maximum scale for drawing commands which follow.

Applicability: All drawing commands except *NullInstruction*

Id*[:id]*

Sets an identifier for drawing commands which follow. When no parameters are present, resets to the default state of no identifier.

id An identifier for drawing commands which follow

Applicability: All drawing commands

Parent*[:id]*

Visibility of drawing commands which follow is dependent on the referenced drawing command(s). If no referenced drawing command is executed during rendering then the dependent drawing commands should not be executed.

In order to express cross-feature dependencies, the referenced drawing command(s) may be associated with a feature instance other than the current feature instance; examine all drawing commands for all feature instances when determining the parent drawing command(s).

Execution of referenced (parent) drawing commands can be affected by many aspects of the visualization process including viewing group settings, display plane visibility, line suppression, scale minimum/maximum, date dependency, hover status and dependencies of the parent drawing command.

When no parameters are present, resets to the default state of no parent dependency.

id The identifier of the parent drawing command(s)

Applicability: All drawing commands except *NullInstruction*

Hover:*hover*

Specifies whether visibility of drawing commands which follow is dependent on hover-over. OEM implementation of support for this feature is optional.

hover

true Visibility is dependent on hover-over

false Visibility is not dependent on hover-over

Applicability: All drawing commands except *NullInstruction*

9a-11.2.2.2 Transform Commands

Transform commands apply transformations to elements, such as symbols, rendered by applicable drawing commands which follow.

Table 9a-8 – Transform Commands

Command	Parameters	Type	Initial State	Part 9 Reference
LocalOffset	xOffsetMM	double	0	9-12.2.2.7
	yOffsetMM	double	0	
LinePlacement	linePlacementMode	string	Relative	9-12.3.1.5
	offset	double	0.5	
	endOffset	double	nil	
	visibleParts	boolean	false	
AreaPlacement	areaPlacementMode	string	VisibleParts	9-12.3.1.6
AreaCRS	areaCRSType	string	GlobalGeometry	9-12.5.1.9
Rotation	rotationCRS	string	PortrayalCRS	9-12.2.2.7 9-12.3.1.1 9-12.4.1.4 9-12.6.3.5
	rotation	double	0	
ScaleFactor	scaleFactor	double	1.0	9-12.2.2.7 9-12.3.1.1 9-12.4.1.4

LocalOffset:*xOffsetMM,yOffsetMM*

Specifies an offset from the geographic position using the Local CRS to be applied to subsequent drawing commands.

Applicability: *PointInstruction, SymbolFill, TextInstruction*

LinePlacement:*linePlacementMode,offset[,endOffset][,visibleParts]*

Specifies the placement along a line for symbols or text output by subsequent drawing commands.

linePlacementMode

Relative *offset* is in homogenous coordinates, 0 for the start and 1 for the end of the curve.

Absolute *offset* specifies the distance from the start of the curve.

offset The offset from the start of the curve. If *endOffset* is also specified this indicates the start point of any *TextInstruction* command.

endOffset If present, indicates the text drawn by any *TextInstruction* is depicted along the linear geometry.

If present and contains a value, further indicates the end point of any *TextInstruction* command. Horizontal alignment has no effect in this case.

visibleParts If present and true, indicates the placement applies to each visible part of the curve.

Applicability: *PointInstruction, TextInstruction*

AreaPlacement:*areaPlacementMode*

Specifies the placement within an area for symbols or text output by subsequent drawing commands.

areaPlacementMode – one of:

VisibleParts The symbol or text is to be placed at a representative position in each visible part of the surface.

Geographic The symbol or text is to be placed at a representative position of the geographic object.

Applicability: *PointInstruction, TextInstruction*

AreaCRS:*areaCRSType*

Specifies how fill patterns output by subsequent drawing commands are anchored.

areaCRSType – one of:

Global The anchor point is consistent with a location on the drawing device; for example, starting with the corner of the screen. As the screen pans the pattern will appear to shift/move through the object on screen.

LocalGeometry The anchor point is consistent with the local geometry of the object being depicted, for example the upper left corner of the object. Patterns of adjacent objects may not match.

GlobalGeometry The anchor point of the fill pattern is defined at a common location such that patterns remain consistent relative to all area objects.

Applicability: *AreaFillReference, PixmapFill, SymbolFill, HatchFill, TextInstruction*

Rotation:*rotationCRS,rotation*

Specifies the rotation angle for symbols or text output by subsequent drawing commands.

rotationCRS – one of:

GeographicCRS A geographic CRS with axis latitude and longitude measured in degrees. *rotation* is defined as clockwise from the true north direction.

PortrayalCRS A Cartesian coordinate system with the y-axis pointing upwards. *rotation* is defined in degrees clockwise from the positive y-axis.

LocalCRS A Cartesian coordinate system originated at a local geometry. *rotation* is in degrees clockwise from the positive y-axis.

LineCRS A none-Cartesian coordinate system where the x-axis is following the geometry of a curve and the y-axis is perpendicular to the x-axis (positive to the left of the x-axis).

Units on the axes and for distances are millimetres. Angles are measured in degrees clockwise from the positive y-axis.

See Part 9 clause 9-12.2.2.7 for details.

Applicability: *PointInstruction, SymbolFill, TextInstruction, CoverageFill*

ScaleFactor:*scaleFactor*

Specifies a scale factor to be applied to symbols or text output by subsequent drawing commands.

Applicability: *PointInstruction, SymbolFill, TextInstruction, CoverageFill*

9a-11.2.2.3 Line Style Commands

Line style commands create linestyles which may be referenced by subsequent drawing commands. These commands are part of the functionality of the *LineStyles* package described in Part 9 clause 9-12.4.

Table 9a-9 – LineStyle Commands

Command	Parameters	Type	Initial State	Part 9	Notes
Dash	start	double	-	9-12.4.1.3	Units: millimetres
	length	double	-		
LineSymbol	reference	string	-	9-12.4.1.4	
	position	double	-		
	rotation	double	0		
	crsType	CRSType	LocalCRS		
	scaleFactor	double	1.0		

LineStyle	name	string	-	9-12.4.1.1 9-12.4.1.2	
	intervalLength	double	-		
	width	double	-		
	token	string	-		
	transparency	double	0		
	capStyle	string	Butt		
	joinStyle	string	Miter		
	offset	double	0.0		

Dash:*start,length*

Specifies a dash pattern for a single subsequent *LineStyle* command. Can be repeated to specify that multiple dash patterns apply to the single *LineStyle* command.

NOTE: This command does not set the state for any drawing command; it only sets the state for the *LineStyle* command.

start The start of the dash measured from the start of the line along the x-axis of the line CRS (units in millimetres).

length The length of the dash along the x-axis of the line CRS (units in millimetres).

Applicability: *LineStyle*

LineSymbol:*reference,position[,rotation[,crsType[,scaleFactor]]]*

Specifies the use of a symbol for a single subsequent *LineStyle* command. Can be repeated to specify that multiple symbols apply to the *LineStyle* command.

reference A reference to an external definition of the symbol graphic. This refers to an identifier of a portrayal catalogue item.

position The position of the symbol measured from the start of the repeating interval, along the x-axis of the line CRS (units in millimetres).

rotation The rotation angle of the symbol.

crsType The type of the CRS where the symbol has to be transformed. Possible values are LocalCRS, LineCRS and PortrayalCRS.

scaleFactor The scale factor of the symbol.

Applicability: *LineStyle*

LineStyle:*name,intervalLength,width,token[,transparency[,capStyle[,joinStyle[,offset]]]]*

Creates a named linestyle for use by subsequent drawing commands. May be preceded by zero or more *Dash* and/or *LineSymbol* commands which apply to the linestyle. If no *Dash* commands precede the *LineStyle* command, a solid line is created.

name A name assigned to the linestyle and used to reference the linestyle from a *LineInstruction*. In the event of a name collision between a Portrayal Catalogue linestyle and a *LineStyle* command, the *LineStyle* command takes precedence.

intervalLength The length of a repeating interval of the line style along the x-axis of the line CRS (units in mm). Can be omitted if a solid is being defined.

width Pen width in mm used to draw this line style.

token Specifies the colour used to draw this line style.

transparency Specifies the transparency used to draw this line style.

capStyle The decoration that is applied where a line segment ends. One of *Butt*, *Square*, or *Round*. See Part 9 clause 9-12.4.1.8 *CapStyle*.

joinStyle The decoration that is applied where two line segments meet. One of *Bevel*, *Miter*, or *Round*. See part 9 clause 9-12.4.1.7 *JoinStyle*.

offset An offset perpendicular to the direction of the line. The value refers to the y-axis of the line CRS (positive to the left, millimetres).

Applicability: *LineInstruction, LineInstructionUnsuppressed, HatchFill*

9a-11.2.2.4 Text Style Commands

Text style commands modify the appearance of text drawn by subsequent drawing commands.

Table 9a-10 – Text Style Commands

Command	Parameters	Type	Initial State	Part 9	Notes
FontColor	token	string	""	9-12.6.3.8	Opaque
	transparency	double	0	9-12.2.2.3	
FontBackgroundColor	token	string	""	9-12.6.3.8	Transparent
	transparency	double	1	9-12.2.2.3	
FontSize	bodySize	double	10	9-12.6.3.8	
FontProportion	proportion	string	Proportional	9-12.6.3.11	
FontWeight	weight	string	Medium	9-12.6.3.10	
FontSlant	slant	string	Upright	9-12.6.3.9	
FontSerifs	serifs	boolean	false	9-12.6.3.2	
FontUnderline	underline	boolean	false	9-12.6.3.12	
FontStrikethrough	strikethrough	boolean	false	9-12..6.3.12	
FontUpperline	upperline	boolean	false	9-12.6.3.12	
FontReference	fontReference	string	""	9-12.6.3.3	
TextAlignHorizontal	horizontalAlignment	string	Start	9-12.6.3.14	
TextAlignVertical	verticalAlignment	string	Bottom	9-12.6.3.13	
TextVerticalOffset	verticalOffset	double	0	9-12.6.3.8	

FontColor:*token[,transparency]*

Specifies the colour and transparency for glyphs drawn by subsequent drawing commands.

Applicability: *TextInstruction*

FontBackgroundColor:*token[,transparency]*

Specifies the colour and transparency used to fill the rectangle surrounding text drawn by subsequent drawing commands.

Applicability: *TextInstruction, CoverageFill*

FontSize:*bodySize*

Specifies the size in points for text drawn by subsequent drawing commands.

Applicability: *TextInstruction, CoverageFill*

FontProportion:*proportion*

Specifies a font proportion to be used for text drawn by subsequent drawing commands.

proportion – one of:

MonoSpaced A font where all typefaces have the same width should be selected. Also known as 'typewriter' fonts.

Proportional A font where each typeface can have a different width should be selected.

Applicability: *TextInstruction, CoverageFill*

FontWeight:*weight*

Specifies the font thickness for text drawn by subsequent drawing commands.

weight – one of:

Light Typefaces are depicted as thin (standard).

Medium Typefaces are depicted thicker than *Light*, but not as thick as *Bold*.

Bold Typefaces are depicted more prominently (**Bold**).

Applicability: *TextInstruction, CoverageFill*

FontSlant:*slant*

Specifies the slant to be used for text drawn by subsequent drawing commands.

slant – one of:

Upright Typefaces are upright.

Italics Typefaces are slanted to the right.

Applicability: *TextInstruction, CoverageFill*

FontSerifs:*serifs*

Specifies whether the font used for text drawn by subsequent drawing commands should contain serifs.

Applicability: *TextInstruction, CoverageFill*

FontUnderline:*underline*

Specifies whether text drawn by subsequent drawing commands should be underlined.

Applicability: *TextInstruction*

FontStrikethrough:*strikethrough*

Specifies whether text drawn by subsequent drawing commands should be depicted with a line through the center of the text.

Applicability: *TextInstruction*

FontUpperline:*upperline*

Specifies whether text drawn by subsequent drawing commands should be depicted with a line above the text.

Applicability: *TextInstruction*

FontReference:*fontReference*

Specifies text drawn by subsequent drawing commands should be depicted using the specified font from the Portrayal Catalogue. *fontReference* is the identifier for the external file within the Portrayal Catalogue.

Applicability: *TextInstruction*

TextAlignHorizontal:*horizontalAlignment*

Specifies the text placement relative to the anchor point in the horizontal direction for subsequent drawing commands.

horizontalAlignment – one of:

Start	The anchor point is at the start of the text.
Center	The anchor point is at the (horizontal) centre of the text.
End	The anchor point is at the end of the text.

Applicability: *TextInstruction*

TextAlignVertical:*verticalAlignment*

Specifies the text placement relative to the anchor point in the vertical direction for subsequent drawing commands.

verticalAlignment – one of:

Top	The anchor point is at the top of the text.
Center	The anchor point is at the (vertical) centre of the text.
Bottom	The anchor point is at the bottom of the em text.

Applicability: *TextInstruction*

TextVerticalOffset:*verticalOffset*

Specifies the vertical offset in mm above the anchor point of the text drawn by subsequent *TextInstruction* commands. Used to generate subscripts or superscripts.

Applicability: *TextInstruction*

9a-11.2.2.5 Colour Override Commands

Colour override commands modify the colour of symbols and pixmaps drawn by subsequent drawing commands.

Table 9a-11 – Colour Override Commands

Command	Parameters	Type	Initial State	Part 9	Notes
OverrideColor	colorToken	string	N/A	9-12.2.2.6 9-12.3.1.2	
	colorTransparency	double	N/A		
	overrideToken	string	N/A		
	overrideTransparency	double	N/A		
OverrideAll	token	string	N/A	9-12.2.2.5 9-12.3.1.1	
	transparency	double	N/A		
ClearOverride					

OverrideColor:*colorToken,colorTransparency,overrideToken,overrideTransparency*

Specifies an override colour which should be used to replace the original colour in a symbol or pixmap rendered via a drawing command. This command can be issued multiple times to specify more than one colour substitution.

Applicability: *PointInstruction, AreaFillReference, PixmapFill, SymbolFill*

OverrideAll:*token,transparency*

Substitutes all non-transparent colours with the given colour. This command supercedes any *OverrideColor* commands.

Applicability: *PointInstruction, AreaFillReference, PixmapFill, SymbolFill*

ClearOverride

Removes all colour substitutions.

Applicability: *PointInstruction, AreaFillReference, PixmapFill, SymbolFill*

9a-11.2.2.6 Geometry Commands

All drawing commands defined in clause 9a-11.2.1 reference geometries. Normally, this is the geometry of the feature (analogous to Part 9 clause 9-11.2.2 *DrawingInstruction::featureReference*). The host determines the feature's geometry using the feature reference provided when drawing instructions are returned from the portrayal via *HostPortrayalEmit* as described in clause 9a-14.2.1. The geometry commands defined in this section allow the normal behaviour to be overridden.

One method of overriding the normal behaviour is to constrain drawing commands so that they reference either individual geometric elements of a feature; or any other geometries defined in the dataset (analogous to Part 9 clause 9-11.2.2 *DrawingInstruction::spatialReference*).

The second method of overriding the normal behaviour is to create an augmented geometry (Part 9 clause 9-11.1.13 Augmented Geometry) using a geometry command. Augmented geometry is used when the spatial to be referenced is not present in the dataset. Augmented geometry created by a geometry command will be used by subsequent drawing commands, overriding the features geometry.

This Part does not define separate augmented drawing instructions as in Part 9. Instead, all drawing commands are to use augmented geometry whenever augmented geometry is available.

To deterimine the geometry to be used by a drawing command:

- If an augmented geometry command precedes the drawing command, the most recently defined augmented geometry should be used.
- Otherwise, if the spatial references list is not empty, the drawing command is applied to each spatial reference.
- Otherwise, the features geometry should be used.

To implement augmented paths, the host should maintain a segment list into which the geometries created by the *Polyline*, *Arc3Points*, *ArcByRadius* and *Annulus* commands are placed. This list maintains the order in which the geometries are created.

Applied geometry commands are removed via the *ClearGeometry* command, which also clears the segment list. Using *ClearGeometry* allows switching between referencing the features geometry, augmented geometry, and spatial references.

The geometry commands are listed in the table below. The type *point* indicates a pair of doubles are passed as parameters.

Table 9a-12 – Geometry Commands

Command	Parameters	Type	Initial State	Part 9	Notes
SpatialReference	reference	string	-	9-11.2.4	
	forward	boolean	true		
AugmentedPoint	crs	CRSType	-	9-11.2.13	
	x	Point	-		
	y		-		
AugmentedRay	crsDirection	CRSType	-	9-11.2.15	
	direction	double	-		
	crsLength	CRSType	-		
	length	double	-		
AugmentedPath	crsPosition	CRSType	-	9-11.2.16	
	crsAngle	CRSType	-	9-11.2.16	
	crsDistance	CRSType	-	9-11.2.16	
Polyline	point1	Point[]	-	9-12.2.2.11	

	…				
	pointN				
Arc3Points	startPointX	Point	-	9-12.2.2.13	
	startPointY				
	medianPointX	Point	-		
	medianPointY				
	endPointX	Point	-		
	endPointY				
ArcByRadius	centerX	Point	-	9-12.2.2.14	
	centerY				
	radius	double	-		
	startAngle	double	0		
	angularDistance	double	360		
Annulus	centerX	Point	-	9-12.2.2.15	
	centerY				
	outerRadius	double	-		
	innerRadius	double	outerRadius		
	startAngle	double	0		
	angularDistance	double	360		
ClearGeometry	-	-	-	-	

SpatialReference:*reference[,forward]*

Specifies a reference to the spatial type components of the feature that defines the geometry used for the depiction of drawing commands which follow. Not used when the entire geometry of the feature should be depicted. Each time this command is called, a new spatial reference is added to the spatial references list maintained by the host. The spatial references list can be cleared by calling *ClearGeometry.*

reference The identifier of the spatial type as defined in Part 13 clause 13-8.

forward If true the spatial object is used in the direction in which it is stored in the data. Only applies to curves and should be ignored for all other spatial types.

Applicability: All drawing commands

AugmentedPoint:*crs,x,y*

Specifies a position. Clears any active *AugmentedRay* and *AugmentedPath* instructions.

crs – one of:

GeographicCRS A geographic CRS with axis latitude and longitude measured in degrees.

PortrayalCRS A Cartesian coordinate system with the y-axis pointing upwards. Units on the axes and for distances are millimetres.

LocalCRS A Cartesian coordinate system originated at a local geometry. Units on the axes and for distances are millimetres.

x,y Coordinates of the point.

Applicability: *PointInstruction, TextInstruction, NullInstruction*

AugmentedRay:*crsDirection,direction,crsLength,length*

Augments the geometry of a point feature. Specifies a line from the position of the point feature to another position. The position is defined by the direction and the length attributes. Clears any active *AugmentedPoint* and *AugmentedPath* instructions.

If *crsDirection* is *PortrayalCRS* or *LocalCRS* then *crsLength* must be *PortrayalCRS* or *LocalCRS.* Similarly, if *crsLength* is *GeographicCRS* then *crsDirection* must be *GeographicCRS.*

crsDirection and *crsLength* – each one of:

GeographicCRS Angles are defined clockwise from the true north direction. Distances will be measured in metres.

PortrayalCRS A Cartesian coordinate system with the y-axis pointing upwards. Units on the axes and for distances are millimetres. Angles are measured in degrees clockwise from the positive y-axis.

LocalCRS A Cartesian coordinate system originated at a local geometry. Units on the axes and for distances are millimetres. Angles are measured in degrees clockwise from the positive y-axis.

direction The direction of the ray relative to the CRS specified.

length The length of the ray in units depending on the CRS specified.

Applicability: *LineInstruction, LineInstructionUnsuppressed, TextInstruction, NullInstruction*

AugmentedPath:*crsPosition,crsAngle,crsDistance*

Instructs the host to gather all segments previously created by *Polyline*, *Arc3Points*, *ArcByRadius* and *Annulus* commands and group them as a single augmented geometry. The host must then clear the segment list. Clears any active *AugmentedPoint* and *AugmentedRay* instructions.

To implement an augmented path, the host must maintain a segment list. Each call to *Polyline*, *Arc3Points*, *ArcByRadius* and *Annulus* results in the host placing the geometry on the segment list. These items taken in order they are added to the segment list define the augmented path.

The CRS is specified separately for positions, angles and distances.

crsPosition, crsAngle and *crsDistance* – each one of:

GeographicCRS A geographic CRS with axis latitude and longitude measured in degrees. Angles are defined clockwise from the true north direction. Distances will be measured in metres.

PortrayalCRS A Cartesian coordinate system with the y-axis pointing upwards. Units on the axes and for distances are millimetres. Angles are measured in degrees clockwise from the positive y-axis.

LocalCRS A Cartesian coordinate system originated at a local geometry. Units on the axes and for distances are millimetres. Angles are measured in degrees clockwise from the positive y-axis.

Applicability: All drawing commands except *PointInstruction*

Polyline:*positionXstart,positionYstart,positionXto,positionYto[,positionXto,positionYto…]*

Instructs the host to add a polyline to the segment list.

positionXstart,positionYstart,positionXto,positionYto Coordinates of the segments of the polyline.

Applicability: *AugmentedPath*

Arc3Points:*startPointX,startPointY,medianPointX,medianPointY,endPointX,endPointY*

Instructs the host to add an arc defined by three points to the segment list.

startPointX,startPointY The point where the arc starts.

medianPointX,medianPointY An arbitrary point on the arc.

endPointX,endPointY The point where the arc ends.

Applicability: *AugmentedPath*

ArcByRadius:*centerX,centerY,radius[,startAngle,angularDistance]*

Instructs the host to add an arc defined by a radius to the segment list.

centerX,centerY The centre of the arc.

radius The radius of the circle.

startAngle,angularDistance The sector defining where the arc starts and ends. If not present the arc is a full circle.

Applicability: *AugmentedPath*

Annulus:*centerX,centerY,outerRadius[,innerRadius[,startAngle,angularDistance]]*

Instructs the host to add an annulus to the segment list. An annulus is a ring-shaped region bounded by two concentric circles. It can optionally be bounded by two radii of the circle.

Note that the presence of *startAngle* and *angularDistance* parameters does not imply that *innerRadius* must be present. The following is a valid command: `Annulus:0,1,2.34,,56,78`

centerX,centerY The centre of the annulus.

outerRadius The radius of the larger circle.

innerRadius The radius of the smaller circle. If not present the segment describes a sector of a circle.

startAngle,angularDistance The sector of an annulus segment.

Applicability: *AugmentedPath*

ClearGeometry

Clears any preceding geometry commands and empties the segment and spatial references lists.

Applicability: *AugmentedPath, SpatialReference*

9a-11.2.2.7 Coverage Commands

Coverage commands define lookup entries which are referenced by the *CoverageFill* drawing command. These commands are part of the functionality of the *Coverage* package described in Part 9 clause 9-12.7. The coverage commands are listed in Table 9a-13 below.

Table 9a-13 - Coverage Commands

Command	Parameters	Type	Initial State	Part 9	Notes
NumericAnnotation	decimals	integer	-	9-12.7.4.4	
	championChoice	ChampionChoice	-		
	buffer	double	0		
SymbolAnnotation	symbolRef	string	-	9-12.7.4.5	
	rotationAttribute	string	-		
	scaleAttribute	string	-		
	rotationCRS	CRSType	PortrayalCRS		
	rotationOffset	double	0		
	rotationFactor	double	1		
	scaleFactor	double	1		
CoverageColor	startToken	string	-	9-12.7.4.3	
	startTransparency	dDouble	0		

	endToken	string	-		
	endTransparency	double	0		
	penWidth	double	0		
LookupEntry	label	string	-	9-12.7.4.2 1-4.5.3.4	
	lower	double	-		
	upper	double	-		
	closure	S100_IntervalType	-		

NumericAnnotation:*decimals,championChoice[,buffer]*

Specifies the numeric representation of a coverage instruction. When executing the *CoverageFill* drawing command, the numeric value should be drawn using the currently defined font. However, instead of using the font colour set by *FontColor, CoverageColor* should be used.

decimals Number of decimal digits to show in subscript.

championChoice – one of:

Largest Display the largest value in case of collision.

Smallest Display the smallest value in case of collision.

buffer Buffer to apply for collision detection in portrayal units.

Applicability: *LookupEntry*

SymbolAnnotation:*symbolRef,rotationAttribute,scaleAttribute[,rotationCRS,rotationOffset[,rotationFactor[,scaleFactor]]]*

Specifies the symbol representation of a coverage instruction.

symbolRef The symbol from the Portrayal Catalogue to draw.

rotationAttribute The attribute code of the Coverage Attribute to use for the symbol rotation value.

scaleAttribute The attribute code of the Coverage attribute to use for scaling the symbol size.

rotationCRS Specifies the coordinate reference system for the rotation.

rotationOffset Used to adjust the ‘rotationAttribute’ value by addition before applying. This offset is applied after *rotationFactor.* If no *rotationAttribute* is given, this value represents the rotation value to apply to the symbol. A value of 0 indicates no adjustment.

rotationFactor Used to adjust the ‘rotationAttribute’ value by multiplication before applying. This factor is applied before *rotationOffset.* A value of 1 indicates no adjustment.

scaleFactor Used to adjust the ‘scaleAttribute’ value by multiplication before applying. A value of 1 indicates no adjustment.

EXAMPLE: Assume a coverage has wind speed and direction attributes and the portrayal wishes to draw an arrow showing wind direction and whose length is proportion to the wind speed. In this example the wind direction indicates the compass direction of where the wind is coming from and the portrayal wants to indicate the direction the wind is blowing towards. Additionally, the portrayal wants a 20 knot wind speed to be indicated by drawing the arrow at its normal scale. In this case the portrayal needs to rotate the arrow by 180 degrees and scale the arrow by 1/20. The following commands could be used to accomplish the portrayal of the arrow:

```
SymbolAnnotation:ARROW,windDirection,windSpeed,PortrayalCRS,180,1.0,0.05;
LookupEntry:Wind,0,360,closedInterval;
CoverageFill:windDirection
```

Applicability: *LookupEntry*

CoverageColor:*startToken,startTransparency[,endToken,endTransparency][,penWidth]*

Specifies the colour range to use for a coverage instruction. If *endToken* and *endTransparency* are not specified, then a single colour is used.

startToken,startTransparency The colour to assign to the matching range or to use as start point in a colour ramp when 'endColor' is defined.

endToken,endTransparency If given, the colour to use as the stopping point in a coluor ramp. The range of values is spread linearly across the range of colours from 'startColor' to 'endColor' to produce a gradient effect.

The start and end colours shall be converted to the CIE xyL colour space and each component (that is x, y, L and transparency) shall be interpolated individually before converting back to sRGB for display.

penWidth Pen width to apply for dot colour used for discrete points.

Applicability: *LookupEntry*

LookupEntry:*label,lower,upper,closure*

Creates a lookup entry for use by a single subsequent *CoverageFill* drawing command. This instruction is used to associate preceding *NumericAnnotation, SymbolAnnotation* and *CoverageColor* commands with a single lookup table entry.

NOTE: To support associating alerts with coverage values which differ from portrayal, preceding *NumericAnnotation*, *SymbolAnnotation*, or *CoverageColor* commands are not required.

NOTE: subsequent *LookupEntry* commands require redefinition of *NumericAnnotation*, *SymbolAnnotation*, and *CoverageColor*; for example the state of the other coverage commands should be reset after processing *LookupEntry*.

label String used as a display label or legend field.

lower Lower value of lookup range.

upper Upper value of lookup range.

Closure Interval closure for range. See Part 1 clause 1-4.5.3.4.

Applicability: *CoverageFill*

9a-11.2.2.8 Time Commands

Time commands apply time intervals to drawing commands which follow. A time interval is described by a start and end date and time. The time commands are listed in Table 9a-14.

Table 9a-14 - Time Commands

Command	Parameters	Type	Initial State	Clause	Notes
Date	begin	string	-∞	1-4.5.3.10	
	end	string	∞		
Time	begin	string	-∞	1-4.5.3.10	
	end	string	∞		
DateTime	begin	string	-∞	1-4.5.3.10	
	end	string	∞		
TimeValid	closure	S100_IntervalType	-	1-4.5.3.11 9-11.2.2	
ClearTime	-	-	-	-	

Date:*[begin][,end]*

Specifies a start and/or end date for a time interval. One of *begin* or *end* must be present.

begin Start of the interval. A *S100_TruncatedDate*, see Part 1 table 1-2.

end End of the interval. A *S100_TruncatedDate*, see Part 1 table 1-2.

Applicability: *TimeValid*

Time:*[begin][,end]*

Specifies a start and/or end time for a time interval. One of *begin* or *end* must be present.

begin Start of the interval. A *Time,* see Part 1 table 1-2.

end End of the interval. A *Time*, see Part 1 table 1-2.

Applicability: *TimeValid*

DateTime:*[begin][,end]*

Specifies a start and/or end date and time for a time interval. One of *begin* or *end* must be present.

begin Start of the interval. A *DateTime,* see Part 1 table 1-2.

end End of the interval. A *DateTime,* see Part 1 table 1-2.

Applicability: *TimeValid*

TimeValid*[:closure]*

Creates a time interval which applies to drawing commands which follow. Intervals accumulate until cleared via the *ClearTime* command.

If any interval coincides with the viewing date of the portrayal (or other appropriate selector) then the drawing command and its associated information (for example, alerts) are valid, otherwise the command and its associated information are disabled.

The start and/or end values of the time interval are set by preceding *Date*, *Time*, or *DateTime* commands.

closure A string specifying an *S100_IntervalType.* See Part 1 clause 1-4.5.3.4. If not specified, defaults to *closedInterval*.

NOTE: A single-value interval is specified with *begin* = *end* and *closure* = *closedInterval*.

NOTE: The host must clear the state set by preceding *Date*, *Time*, and *DateTime* commands upon completion of this command.

Applicability: All drawing commands

ClearTime

Clears all accumulated time intervals and all state set by preceding time commands.

Applicability: All time commands, all drawing commands

9a-11.2.2.9 Alert Commands

Alert commands associate alert information with the geometry of a drawing instruction.

Table 9a-155 – Alert Commands

Command	Parameters	Type	Initial State	Part 9	Notes
AlertReference	alertReference	string	-	9-11.2.2 9-11.2.5	
	plan	string	-		
	monitor	string	-		

AlertReference*[:alertReference[,plan[,monitor]]]*

Specifies alert information to be associated with the geometry of drawing instructions which follow.

alertReference	A reference to an alert in the alert catalogue. If not provided, clears any previously applied alert reference.
plan	The viewing group the alert highlight is assigned to when active in route planning. If not provided, no viewing group is assigned and the highlight cannot be disabled.
monitor	The viewing group the alert highlight is assigned to when active in route monitoring. If not provided, no viewing group is assigned and the highlight cannot be disabled.

Applicability: All drawing commands

9a-12 Symbol Definitions

The symbol definitions described in Part 9 clause 9-12 are implemented within the Model of the Drawing Instructions (see clause 9a-11.2).

9a-13 The Portrayal Library

There is no change to the organization structure of the portrayal library as defined in Part 9 clause 9-13.2. The "Rules" folder XSLT contents of Part 9 clause 9-13.2 are replaced with Lua script files. *FileType:rules d*escribed in Part 9 clause 9-13.3.32 is used to identify each of the Lua script files.

9a-14 Portrayal Domain Specific Functions

The Lua portrayal is an instance of a Part 13 scripting domain. The functions described below are specific to this scripting domain; they are domain specific functions to be used in conjunction with the standard functions detailed in Part 13.

9a-14.1 Portrayal Domain Specific Catalogue Functions

The functions listed on the following clauses are implemented within the Portrayal Catalogue rule files. They can be called by the host, and augment the standard catalogue functions described in Part 13.

9a-14.1.1 Boolean PortrayalMain(String[] featureIDs)

Return Value:

true Portrayal completed successfully.

false Portrayal was terminated by the host (host returned false from *HostPortrayalEmit*).

Parameters:

featureIDs: String[]

An array containing the IDs of the features for which to generate drawing instructions. If this parameter is nil (or missing), the portrayal will generate drawing instructions for all feature instances in the dataset.

Remarks:

This function is called by the host to start the portrayal process for a dataset instance. Subsequently, the portrayal scripts will repeatedly call *HostPortrayalEmit*, providing the host with the drawing instructions for each feature instance portrayed.

The function returns once the portrayal scripts have run to completion; an error is thrown; or the host returns false from *HostPortrayalEmit*.

If using a portrayal cache as outlined in clause 9a-5.2.2.1, the host only needs to pass in uncached featureIDs, or featureIDs associated with context parameters whose values have changed.

9a-14.1.2 void PortrayalInitializeContextParameters(ContextParameter[] c*ontextParameters*)

Return Value:

void

Parameters:

contextParameters: ContextParameter[]

An array of ContextParameter objects.

Remarks:

Provides the portrayal scripts with the default value for each portrayal context parameter defined within the Portrayal Catalogue. *PortrayalCreateContextParameter* should be used to create each entry. The host is responsible for retrieving the portrayal context parameters from the Portrayal Catalogue.

9a-14.1.3 ContextParameter PortrayalCreateContextParameter(String *contextParameterId*, String *contextParameterType*, String *defaultValue*)

Return Value:

A ContextParameter storing the *defaultValue* with the *contextParameterId*.

Parameters:

contextParameterId: String

The ID of a portrayal context parameter. Valid IDs are defined in the Portrayal Catalogue.

contextParameterType: String

The type of the portrayal context parameter. Valid values are *Boolean, Integer, Real, Text* and *Date*.

defaultValue: String

The default value for the portrayal context parameter. This value is encoded as described in Part 13 clause 13-8.1.

Remarks:

Creates a ContextParameter object for use within the scripting environment.

9a-14.1.4 void PortrayalSetContextParameter(String *contextParameterId*, String *value*)

Return Value:

void

Parameters:

contextParameterId: String

The ID of a portrayal context parameter.

value: String

The new value for the portrayal context parameter. This value is encoded as described in Part 13 clause 13-8.1.

Remarks:

Allows the host to modify the value of a portrayal context parameter. The context parameter must be created via *PortrayalInitializeContextParameters* prior to being modified.

9a-14.2 Portrayal Domain Specific Host Functions

The host must implement the function described in the following clause in order to support portrayal. This function is called from the portrayal domain specific catalogue functions, and augments the standard host functions described in Part 13.

9a-14.2.1 Boolean HostPortrayalEmit(String *featureID*, String *drawingInstructions*, String *observedParameters*)

Return Value:

True Continue script processing. The portrayal engine will continue to process feature instances.

False Terminate script processing. No additional feature instances will be processed by the portrayal engine.

Parameters:

featureID: String

Used by the host to uniquely identify a feature instance.

drawingInstructions: String

All of the drawing instructions generated for the feature instance identified by *featureID*. This string is in Data Exchange Format (DEF) as described in Part 13.

observedParameters: String

The context parameters that were observed during the generation of the drawing instructions for this feature. This string is in DEF.

Remarks:

This function is called from the Portrayal Catalogue once per feature instance to provide drawing instructions to the host.

The host can optionally use the observed context parameters to perform drawing instruction caching.

S-100 – Part 10a

ISO/IEC 8211 Encoding

Page intentionally left blank

Contents

Page intentionally left blank

10a-1 Scope

The international standard ISO/IEC 8211 - *Specification for a data descriptive file for information interchange*, is a means of encapsulating data; it provides a file based mechanism for the transfer of data. This Part specifies an interchange format to facilitate the moving of files containing data records between computer systems. It defines a specific structure which can be used to transmit files containing data type and data structures specific to S-100.

10a-2 Conformance

This profile conforms to level 2 of ISO 19106:2004.

10a-3 Normative References

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

ISO/IEC 8211:1994, *Specification for a data descriptive file for information interchange Structure implementations*

10a-4 Structural Implementations

10a-4.1 Introduction

This chapter specifies the structure of an exchange set at the record and field levels. It further specifies the contents of the physical constructs required for their implementation as ISO/IEC 8211 data records, fields, and subfields. The grouping of records into ISO/IEC 8211 files is considered application specific and is, therefore, described in the relevant Product Specification. For the encoding only the binary ISO/IEC 8211 format is used.

10a-4.2 Notations used in this clause

The specification of the structure of a record is given as a tree structure diagram which comprises the names, linkages and repetition factors of the physical constructs. The detailed specifications of fields and subfields are given in tabular form. Additionally for each field the Data Descriptive field is given. Those fields are used in the Data Descriptive Record (DDR) of an ISO/IEC 8211 conformal data set.

10a-4.3 Tree structure diagrams

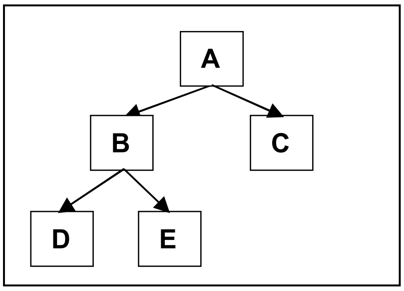


Figure 10a-1

Where A is the root node and parent of node B and node C. Node B is the root of a sub-tree and the parent of nodes D and E.

Nodes are also referred to as the offspring or child of their parents. For example node B is the offspring of node A.

The tree structure diagrams must be interpreted in a pre-ordered traversal sequence (top down, left branch first).

For ease of annotation these diagrams are presented vertically in this standard using ASCII characters. In this notation the above diagram becomes:

```
A
|
|-<r>--B (n): nameOfFieldB
|      |
|      *--D (n*m): nameOfFieldD
|      |
|      *--E (*n): nameOfFieldE
|
|-<r>--C (k\\*n): nameOfFieldC
```

Where:

A, B, C, ...	ISO/IEC 8211 field tags
<r>	r is the sub-tree cardinality (if missing, r=1) possible values: <0.. 1> zero or one <0 .. *> any number including zero <1.. *> at least one
(n)	the number of subfields is n (fixed number)
(n*m)	subfields are stored as an m by n array with m rows and n columns (n subfields are repeated m times)
(*n)	subfields are stored as a n-column table with an arbitrary number of rows (n subfields are repeating)
(k*n)	A concatenation of k subfields and a n-column table (k subfields are followed by n repeating subfields)

The tree structure diagrams define which fields are allowed to be repeated. However, within a record, the degree of repetition of fields will depend on the data that is being encoded. In some cases a particular field may not be required and so will be absent. However, in all cases, the pre-order traversal sequence of a data record will be the same as shown in the generic tree structure diagram for that record type.

10a-4.4 Field Tables

Each table is preceded by a row in bold outline indicating the field name and field tag. The body of the table specifies the subfield names and labels as well as the ISO/IEC 8211. The subfield specification may include a required value or range constraint. The following is an example of a field table using the Data Set Identification field.

Field Tag: **DSID**	Field Name: Data Set Identification

Subfield name	Label	Format	Subfield content and specification
Record name	RCNM	b11	{10} – Data Set Identification
Record identification number	RCID	b14	Range: 1 to 2^{32}-2
Encoding specification	ENSP	A()	Encoding specification that defines the encoding
Encoding specification edition	ENED	A()	Edition of the encoding specification
Product identifier	PRSP	A()	Unique identifier for the data product as specified in the Product Specification
Product edition	PRED	A()	Edition of the Product Specification
Application profile	PROF	A()	Identifier that specifies a profile within the data product
Dataset file identifier	DSNM	A()	The file identifier of the dataset
Dataset title	DSTL	A()	The title of the dataset
Dataset reference date	DSRD	A(8)	The reference date of the dataset Format: YYYMMDD according to ISO 8601
Dataset language	DSLG	A()	The (primary) language used in this dataset

Dataset abstract	DSAB	A()	The abstract of the dataset
Dataset edition	DSED	A()	The edition of the dataset
Dataset topic category	*DSTC	b11	A set of topic categories

Where:

1) **Label** is the ISO/IEC 8211 subfield label, present only in the data descriptive record and required to identify the subfields within a field. A label preceded by "*" signifies that the subfield and the subsequent ones, repeat within the field. This, therefore, indicates the presence of a 2-D array or table for which the subfield labels provide the column headings (the vector labels of a cartesian label).

2) **Format** is the ISO/IEC 8211 binary subfield data format.

10a-4.5 Data formats

Subfield data formats are specified by ISO/IEC 8211. The allowable data formats are as follows:

Format	Data Type	Omitted values	Remark
A(n)	Character Data	If the subfield has a fixed length the subfield will be filled with blanks (space character) If the subfield length is variable only the unit terminator must be encoded	n specified the length of the subfield (number of character) A() indicates a sub field of variable length which must be terminated by the unit delimiter (UT). The encoding of Character Data within this standard must be UTF8 implementation level 1 The appropriate Escape Sequence is: (2/5) (2/15) (4/7) "%/G"
b1w	Unsigned Integer (LSBF) *)	The binary value with all bits set to 1 must be used	w specifies the number of Bytes used Permissible values are: 1,2,4
b2w	Signed Integer (LSBF)	The binary value with all bits set to 1 must be used	w specifies the number of Bytes used Permissible values are: 1,2,4
b48	Signed Floating Point (LSBF)	The value for 'Not A Number' (NaN) must be used	according to IEC 559 or IEEE 754

*) LSBF or "little-endian" is the byte order for multi-byte types. The least significant byte is placed closest to the beginning of a file.

10a-4.6 Data Descriptive fields

Data Descriptive fields are fields of the Data Descriptive Record (DDR) of an ISO/IEC 8211 conformal data file. These fields describe the format of each field in a Data record (DR) of such a file. A Data Descriptive field comprises the Field Control, the Data Field Name, the Array Descriptor, and the Format Controls. More details on Data Descriptive Fields are in ISO/IEC 8211 (1994) Clause 6.4.

Data Descriptive Fields contain non printable characters. In this document they are replaced with graphical symbols as the following table defines:

Character	Code	Graphic
Space	(2/0)	□
UT (Unit Terminator)	(1/15)	▲
FT (Field Terminator)	(1/14)	▼

The Data Descriptive Field is given in a bold text box following the table describing the format of the field.

10a-4.7 Order of records

The order of records will enable the import software to check that a referenced record exists each time it is referenced.

Exists means either:
- The record is inserted in this data set file prior to the record that reference it; or
- It is inserted by the base data set file or an earlier update file and not deleted between the insertion and the record that reference it.

In addition, when a record is going to be deleted it must not be referenced by any other record.

The record order is:
1. Data Set General Information record
2. Data Set Coordinate Reference System record
3. Information Type records (RUIN=Insert) (for the order inside this group see the encoding rules for Information Type records)
4. Point records (RUIN=Insert)
5. Multi Point records (RUIN=Insert)
6. Curve records (RUIN=Insert)
7. Composite Curve records (RUIN=Insert) (for the order inside this group see the encoding rules for Composite Curve records)
8. Surface records (RUIN=Insert)
9. Feature Type records (RUIN=Insert) (for the order inside this group see the encoding rules for Feature Type records)
10. Information Type records (RUIN=Modify)
11. Point records (RUIN=Modify)
12. Multi Point records (RUIN=Modify)
13. Curve records (RUIN=Modify)
14. Composite Curve records (RUIN=Modify)
15. Surface records (RUIN=Modify)
16. Feature Type records (RUIN=Modify)
17. Feature Type records (RUIN=Delete) (reverse order as for Insert)
18. Surface records (RUIN=Delete)
19. Composite Curve records (RUIN=Delete) (reverse order as for Insert)
20. Curve records (RUIN= Delete)
21. Multi Point records (RUIN= Delete)
22. Point records (RUIN= Delete)
23. Information Type records (RUIN= Delete) (reverse order as for Insert)

Note that Product Specifications can omit entries they don't use but not change the order. They might further define a more specific order within each group if the general rule regarding references is not broken.

If several records for the modification of one record are required in one dataset, they must be using increasing record version numbers and the order must be according to these numbers.

10a-4.8 ISO/IEC 8211 file structure

10a-4.8.1 General structure

This clause does not replace ISO/IEC 8211, nor does it give a comprehensive overview of ISO/IEC 8211. The reader is referred to ISO/IEC 8211:1994 for a complete description and explanation.

The clause will give a short overview of ISO/IEC 8211 by explaining those parts of the encapsulation structure which are of relevance to S-100.

ISO/IEC 8211 files are organized by Logical Records (LR), the first record is the Data Descriptive Record (DDR) and all subsequent records are Data Records (DR).

The DDR contains information on the hierarchy and structure of the remaining part of the file. It does not define the semantic of the data.

Each Logical Record (both DDR and DR) contains three basic elements:
- Leader

- Directory
- Field Area

10a-4.8.2 The Leader

The Leader of a Logical Record contains the parameters necessary to read records and to disaggregate the directory into its entries. In addition, the DDR Leader contains a few data descriptive parameters applicable to the entire file. It has a fixed length of 24 bytes.

The first five bytes in any Leader will contain the length of the complete Record in bytes encoded as a decimal number in ASCII representation (for example, a Record of 242 bytes will have a record length entry of “00242”). If a Record has a size of 100000 bytes or larger then the value must be set to “00000”. In this case the software must be able to calculate the Record size from the information in the directory.

10a-4.8.2.1 The DDR Leader

The structure of the DDR Leader:

RP	Len	Entry name	Content
0	5	Record length	Number of bytes in the Record
5	1	Interchange level	"3"
6	1	Leader identifier	"L"
7	1	In line code extension indicator	"E"
8	1	Version number	"1"
9	1	Application indicator	SPACE
10	2	Field control length	"09"
12	5	Base address of Field Area	Start address of Field Area (number of bytes in the Leader and Directory)
17	3	Extended character set indicator	" ! " (SPACE,!,SPACE)
20	4	Entry map	(See below)

The entry map of the DDR Leader:

RP	Sub-entry name	Length	Content
20	Size of field length field	1	Variable "1"-"9" (defined by encoder)
21	Size of field position field	1	Variable "1"-"9" (defined by encoder)
22	Reserved	1	"0"
23	Size of field tag field	1	"4"

10a-4.8.2.2 The DR Leader

The structure of the DR Leader:

RP	Len	Entry name	Content
0	5	Record length	number of bytes in the Record
5	1	Interchange level	SPACE
6	1	Leader identifier	"D"
7	1	In line code extension indicator	SPACE
8	1	Version number	SPACE
9	1	Application indicator	SPACE
10	2	Field control length	2 SPACEs
12	5	Base address of Field Area	Start address of Field Area (number of bytes in the Leader and Directory)

17	3	Extended character set indicator	3 SPACEs
20	4	Entry map	(See below)

The entry map of the DR Leader:

RP	Sub-entry name	Length	Content
20	Size of field length field	1	Variable "1"-"9" (defined by the encoder)
21	Size of field position field	1	Variable "1"-"9" (defined by the encoder)
22	Reserved	1	"0"
23	Size of field tag field	1	"4"

10a-4.8.3 The Directory

The Directory of a Logical Record contains the parameters necessary to identify and locate each field in the Field Area. The Directory consists of repeated Directory entries containing the:

- field tag;
- field length; and
- field position.

The Directory ends with the field terminator (1/14). The field positions are relative to the beginning of the Field Area. The position of the first field following the Directory is 0. The number of bytes used for the three elements (the field entry) is defined by the entry map in the Leader of the Logical Record.

10a-4.8.4 The Field Area

The Field Area is different for the DDR and DR. In the first Record only, the DDR, the Field Area contains data descriptive fields. Each data descriptive field contains information necessary to decode the user data in the Field Area of the DR('s). The fact that the data description is contained in the interchange file makes it possible to exchange data without an external description, though the semantic of the elements is not known. The S-100 Standard and the Product Specifications that use an ISO/IEC 8211 data encoding does contain an external data description used for the exchange of the data. However, the data descriptive fields can only be omitted from the DDR if they are not used in the current file; not because of the existence of an external data description. The data descriptive fields of the DDR form an integral part of an ISO/IEC 8211 conforming file.

The limitation to the used fields is a minimum requirement and other fields may be defined by data descriptive fields in the DDR. However, this adds unnecessary data to the data set and should be avoided.

The Field Area of the DR contains the actual data to be transferred.

10a-4.8.4.1 The Field Area of the DDR

a. Field control field

The first field of the DDR is the field control field. The field tag for the field control field is "0000". The field control field contains a list of field tag pairs. The list defines the hierarchy of all the fields described in the DDR. The list contains pairs of partent/child tags and together with the preorder traversal sequence of the field descriptions in the DDR describes a generic tree structure for the exchange file. The pairs may be placed in the list in any sequence and must be contiguous. The following Figure gives an example of a tree. The set of field tag pairs is HE, EA, EB, HF, HG, GC and GD.

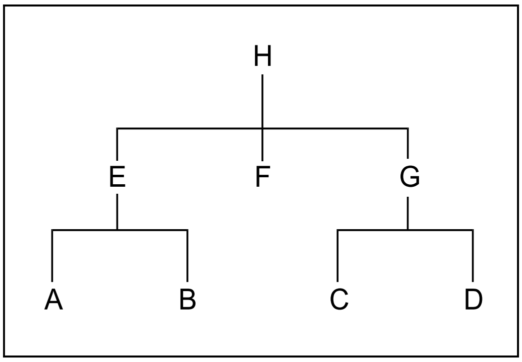


Figure 10a-2

The structure of the field control field is as follows:

Field controls	External file title (Optional for S-100)	UT	List of field tag pairs	FT

The field controls of the field control field are : "0000;&" + 3 SPACEs.

b. Data descriptive fields

The successive fields of the field area contain the data descriptive fields. The data descriptive fields are encoded in the DDR in a preorder traversal sequence. The preorder traversal sequence of the tree shown above is HEABFGCD.

The structure of a data descriptive field is as follows:

Field controls	Field name	UT	Array descriptor	UT	Format controls	FT

The field controls describe the level and data type of the data fields defined by the data descriptive fields. The structure of the field controls is shown in the following Table.

RP	Len	Entry name	Content
0	1	Data structure code	"1" – linear structure "2" – multi-dimensional structure "3" – concatenated structure
1	1	Data type code	"1" – implicit point (integer) "2" – implicit point (float) "6" – mixed data types
2	2	Auxiliary controls	"00"
4	2	Printable graphics	";&"
6	3	Truncated escape sequence	" " (3 SPACEs) – ASCII Encoding ISO 646 "%/G" – UTF8 Encoding implementation level 1

The field name contains the long description of the data fields as defined in the tree structures given in this Part of S-100. The Array descriptor and Format controls define the inner field structure for the associated data fields. Refer to ISO/IEC 8211 for a complete description.

10a-4.8.4.2 The Field Area of the DR's

The data fields in the DR's must be encoded in the preorder traversal sequence as defined in the DDR. The structure of the data fields is defined by the data descriptive fields in the DDR.

10a-4.8.5 An example

The following rather simple example shows an S-100 conformal ISO/IEC 8211 conformal dataset file.

It contains a single feature type (*BuoySafeWater*) with the following attribute set:

Attribute Code	Value	Remarks
buoyShape	“4”	Pillar
colour[1]	“3”	Red
colour[2]	“1”	White
colourPattern	“3”	Diagonal Stripes
featureName[1].language	“eng”	English
featureName[1].name	“Example buoy”	
featureName[2].language	“deu”	German
featureName[2].name	“Beispiel Tonne”	

The Feature Object Id is:

Sub-Field	Integer value	Hexadecimal representation
Producing agency	31868	7C7C
Feature identification number	12345678	00BC614E
Feature identification subdivision	42	002A

The position is:

	Geo position	Integer value	Hexadecimal representation
Latitude	42.42° N	424200000	1948C740
Longitude	12.1234° W	-121234000	F8C61DB0

The example contains non-printable characters and binary codes. They are replaced with the denotation as defined in the following Table:

Character	Code	Denotation	Remarks
Space	Hex 20	□	
UT (Unit Terminator)	Hex 1F	▲	
FT (Field Terminator)	Hex 1E	▼	
Binary code b11	Hex xx	[xx]	
Binary code b12	Hex xxyy	[xxyy]	Due to the LSBF encoding this is equivalent to [yy][xx]
Binary code b14	Hex wwxxyyzz	[wwxxyyzz]	Due to the LSBF encoding this is equivalent to [zz][yy][xx][ww]
Binary code b24	Hex wwxxyyzz	[wwxxyyzz]	For negative numbers the two complement is encoded LSBF
Binary code b48	x.y	[x.y]	Encoded as defined by the IEEE 754 double precision encoding LSBF

DDR Leader

```
011803LE1□0900155□!□3304
```

DDR Directory

```
0000090000DSID132090DSSI118222ATCS044340FTCS047384CSID084431CRSH095515PRID0
71610C2IT055681FRID084736FOID064820ATTR058884SPAS083942▼
```

DDR Field Area

```
0000;&□□□S100Example.000▲DSIDDSSIDSIDATCSDSIDFTCSCSIDCRSHPRIDC2ITFRIDFOIDFR
IDATTRFRIDSPAS▼
```

```
3600;&%/GData□Set□Identification▲RCNM!RCID!ENSP!ENED!PRSP!PRED!PROF!DSNM!DS
TL!DSRD!DSLG!DSAB!DSED\\*DSTC▲(b11,b14,7A,A(8),3A,{b11})▼
```

1600;&□□□Data□Set□Structure□Information▲DCOX!DCOY!DCOZ!CMFX!CMFY!CMFZ!NOIR!NOPN!NOMN!NOCN!NOXN!NOSN!NOFR▲(3b48,10b14)▼

2600;&□□□Attribute□Codes▲*ATCD!ANCD▲(A,b12)▼

2600;&□□□Feature□Type□Codes▲*FTCD!FTNC▲(A,b12)▼

1100;&□□□Coordinate□Reference□System□Record□Identifier▲RCNM!RCID!NCRC▲(b11,b14,b11)▼

1600;&%/GCoordinate□Reference□System□Header▲CRIX!CRST!CSTY!CRNM!CRSI!CRSS!SCRI▲(3b11,2A,b11,A)▼

1100;&□□□Point□Record□Identifier▲RCNM!RCID!RVER!RUIN▲(b11,b14,b12,b11)▼

1100;&□□□2-D□Integer□Coordinate□Tuple▲YCOO!XCOO▲(2b24)▼

1100;&□□□Feature□Type□Record□Identifier▲RCNM!RCID!NFTC!RVER!RUIN▲(b11,b14,2b12,b11)▼

1100;&□□□Feature□Object□Identifier▲AGEN!FIDN!FIDS▲(b12,b14,b12)▼

2600;&%/GAttribute▲*NATC!ATIX!PAIX!ATIN!ATVL▲(3b12,b11,A)▼

2100;&□□□Spatial□Association▲*RRNM!RRID!ORNT!SMIN!SMAX!SAUI▲(b11,b14,b11,2b14,b11)▼

DR 1 Leader (Data Set General Information record)

00321□D□□□□□00065□□□3304

DR 1 Directory

DSID104000DSSI065104ATCS070169FTCS017239▼

DR 1 Field Area

[0A][00000001]S-100□Part□10a▲5.0▲INT.IHO.S-101.1.1▲1.1▲1▲S100Example.000▲S-100□Encoding□example▲20221019EN▲▲1▲[0E][12]▼

[0.0][0.0][0.0][00989680][00989680][00000064][00000000][00000001][00000000][00000000][00000000][00000000][00000001]▼

buoyShape▲[0001]colour▲[0002]colourPattern▲[0003]featureName▲[0004]language▲[0005]name▲[0006]▼

BuoySafeWater▲[0001]▼

DR 2 Leader (Data Set Coordinate Reference System record)

00064□D□□□□□00039□□□2104

DR 2 Directory

CSID070CRSH187▼

DR 2 Field Area

[0F][00000001][01]▼

[01][01][01]WGS□84▲4326▲[02]▲▼

DR 3 Leader (Point Record)

00055□D□□□□□00037□□□1104

DR 3 Directory

PRID90C2IT99▼

DR 3 Field Area

[6E][00000001][0001][01]▼

[1948C740][F8C61DB0]▼

DR 4 Leader (Feature Type Record)

```
00218□D□□□□□00065□□□3304
```

DR 4 Directory

```
FRID011000FOID009011ATTR117020SPAS016137▼
```

DR 4 Field Area

```
[64][00000001][0001][0001][01]▼

[7C7C][00BC614E][002A]▼

[0001][0001][0000][01]4▲
[0002][0001][0000][01]3▲
[0002][0002][0000][01]1▲
[0003][0001][0000][01]3▲
[0004][0001][0000][01]▲
[0005][0001][0005][01]eng▲
[0006][0001][0005][01]Example□buoy▲
[0004][0002][0000][01]▲
[0005][0001][0008][01]deu▲
[0006][0001][0008][01]Beispiel□Tonne▲▼

[6E][00000001][FF][FFFFFFFF][00000000][01]▼
```

10a-5 Common Fields

10a-5.1 Attribute field

10a-5.1.1 Encoding rules

In S-100 attributes can be either simple or complex. Simple attributes have values whereas complex attributes are an aggregation of other attributes, either simple or complex. The following diagram shows an example of a feature type with both simple and complex attributes.

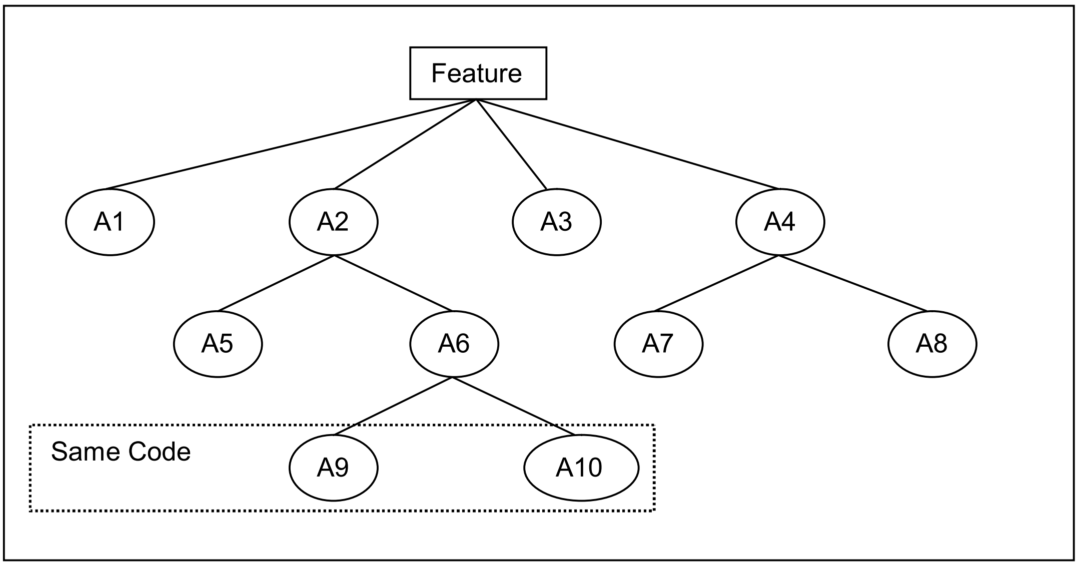


Figure 10a-3

The feature has four attributes: A1, A2, A3, and A4. A1 and A3 are simple attributes; A2 and A4 are complex attributes. A2 comprises two attributes (A5 and A6) where A5 is a simple one and A6 is another complex attribute. A4 and A6 are two complex attributes; both consist of two simple attributes.

Another characteristic of attributes is the cardinality. This indicates how many attributes of the same kind (the same code in a feature catalogue) are used at the same parent. The same parent means that they are all top level attributes or belonging to the same instance of a complex attribute. In the example above A9 and A10 are assumed to have the same code.

With the concept of cardinalities larger than one, an attribute can be seen as an array of attributes. To access an attribute in such an array one needs not only the code of that attribute but also the index of that attribute. Note that the order in such an array may be meaningful and must be maintained by the encoding.

Taking all of the above into account an attribute can be uniquely addressed by three values:

1. The attribute code – encoded with the sub-field NATC;
2. The index of the attribute (starting with 1) – encoded with the sub-field ATIX;
3. The parent of the attribute – encoded with the sub-field PAIX.

The Parent Index (PAIX) is solely used for defining the tree inside the field only, and there is no need to preserve it in an internal system (for example, System Database) structure.

To complete the example above, the following table defines codes and values of the attributes:

Attribute	Code	Attribute Index	Value	Remarks
A1	21	1	Vachon	
A2	22	1		complex
A3	23	1	12	
A4	24	1		complex
A5	25	1	42.0	
A6	26	1		complex
A7	27	1	123	
A8	28	1	Canada	
A9	29	1	17	same code as A10
A10	29	2	43	same code as A9

To encode an attribute a set of five items is necessary: the three mentioned above plus an update instruction and the value of the attribute. To specify the parent of the attribute an index is used. This index points to the n^{th} tuple in the ATTR field starting with 1. The following table shows the encoding of the example:

Index	NATC	ATIX	PAIX	ATIN	ATVL	Remark
1	21	1	0	Insert	Vachon	A1
2	22	1	0	Insert		A2 - complex
3	25	1	2	Insert	42.0	A5
4	26	1	2	Insert		A6 - complex
5	29	1	4	Insert	17	A9
6	29	2	4	Insert	43	A10
7	23	1	0	Insert	12	A3
8	24	1	0	Insert		A4 - complex
9	27	1	8	Insert	123	A7
10	28	1	8	Insert	Canada	A8

Note that here the pre-order traversing is used to define the order of tuples in the field. This keeps all part of a complex attribute together and guarantees that the parent is always stored before the child. The pre-order traversing is defined as follows:

1) Encode the root;

2) Then encode the sub-trees from left to right.

This traversing order is mandatory within this standard.

Note also that the ATIN subfield (Attribute update Instruction) will always be 'Insert' for encoding base data attributes. The other ATIN values (Modify, Delete) are only needed for updating the ATTR field.

All values of attribute are stored as character strings even if the value domain is a numeric type. UTF-8 will be the only encoding allowed in S-100 for such character strings. This allows the encoding of all characters of the first multilingual plane of ISO 10646. There is no other encoding for national character sets necessary.

10a-5.1.2 Updating of the Attribute field

To update an attribute the attribute must be uniquely identifiable and once identified instructions are needed to affect that attribute.

The identifiaction of an attribute is defined by the Numeric Attribute Code (NATC) and the Attribute Index (ATIX).

The Attribute Update Instruction indicates whether an attribute is to be deleted from the field; modified, or inserted. Deletion and modification imply that the attribute exists. Deletion and insertion may change the indices of other attributes in an array of attributes and therefore must be taken into account when the attribute field is updated, thus the ATIX sub-field must use then the corrected value. Instructions must be applied in sequence in order that the indices used are identifying the correct attributes components on subsequent updates.

To demonstrate the updating of attributes the example above should be modified as shown in the following Figure. Note that to identify a node in the attribute tree it is necessary to specify all nodes from the root of the tree. In order to insert the attribute B5 both the nodes A6 and A2 must be encoded before and marked as modified.

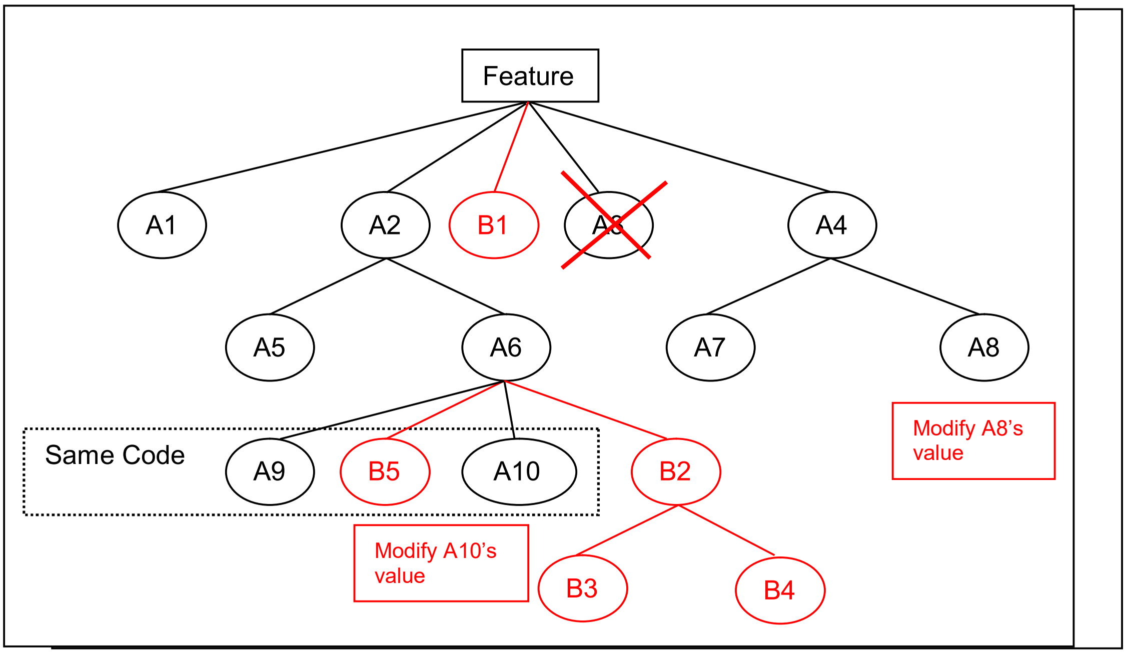


Figure 10a-4

The details are:

Attribute	Code	Attribute Index	Value	Update Instruction	Remarks
B5	29	2	32	Insert	Will change A10's index to 3
A10	29	3	7	Modify	
B2	35	1		Insert	complex
B3	36	1	32	Insert	
B4	37	1	123	Insert	
B1	32	1	abc	Insert	
A3	23	1	12	Delete	
A8	28	1	Germany	Modify	

In order to identify B5, A10 and B2 the entries for A2 and A6 must be inserted. The same is true for A4 (to identify A8). The complete field will look like:

Index	NATC	ATIX	PAIX	ATIN	ATVL	Remark
1	22	1	0	Modify		A2 - complex
2	26	1	1	Modify		A6 - complex
3	29	2	2	Insert	32	B5 - Will increase the ATIX of A10
4	29	3	2	Modify	7	A10 - now with ATIX 3
5	35	1	2	Insert		B2 - complex
6	36	1	5	Insert	22	B3
7	37	1	5	Insert	123	B4
8	32	1	0	Insert	abc	B1
9	23	1	0	Delete		A3
10	24	1	0	Modify		A4 - complex
11	28	1	10	Modify	Germany	A8

Note that in order to delete a complex attribute it will be adequate to delete the root entry of that attribute. For example, to delete A2 only one entry (22, 1, 0, Delete) has to be encoded.

10a-5.1.3 Unknown attribute values

An attribute value is unknown whenever the attribute code is present, but the attribute value is missing (encoded as an empty string).

Updating of the attribute field with a missing attribute value:

- Insert - create the attribute with an initial value of unknown.
- Modify - change the attribute value to unknown.
- Delete - delete the attribute (in this case the attribute value should always be encoded as ‘unknown’).

10a-5.1.4 Encoding of attribute values

The following rules must be used to encode attribute values in the ATVL sub-field:

Attribute Type [Value Type]	Format	Remarks	Examples
S100_GF_TextAttributeType [CharacterString]	Any characters that are supported by the encoding (here - UTF8).	Byte order marks are not permitted since the encoding does not require them (UTF8 is not a multi byte encoding).	whiskey водка
S100_GF_IntegerAttribute [Integer]	Signed Integer numbers of arbitrary length. The numbers must be in the decimal system.	Positive numbers should not use the ‘+’ sign. Negative numbers must have a leading ‘-‘ sign. There must be no whitespaces between the sign and the number. There must be no characters other than the digits ‘0’ to ‘9’ in the number. Non-significant zeros are prohibited.	Valid: 42 -1224566 Not Valid: 0012 123,234 - 12
S100_GF_RealAttribute [Real]	Decimal floating-point numbers in the value domain of double precision numbers (IEEE 64-Bits).	The decimal separator is always a point (‘.’) and there must be no thousands separator used. Positive numbers should not use the ‘+’ sign. Negative numbers must have a leading ‘-‘ sign. The exponential form is permitted. Non-significant zeros are prohibited. If there are only zeros to the right of the decimal point the decimal point should be omitted. The values INF, -INF, and NaN are prohibited.	123.456 -42 1E-5 -2.45E7
S100_BooleanAttributeType [Boolean]	1 (true) or 0 (false)		1 0

S100_EnumerationAttributeType [Positive Integer]	Integer numbers > 0 in the decimal system.	Non-significant zeros are prohibited. The '+' sign should not be used.	
S100_DateAttributeType [Date]	CCYYMMDD Complete representation, basic format according to ISO 8601.	Representation with reduced resolution or truncated representations are prohibited.	20211223
S100_TimeAttributeType [Time]	HHmmss Complete representation, basic format according to ISO 8601.	Optionally a time zone can be added. Fractions of a second are permitted but should be omitted if all zero.	173500 183942+Z 201700-0500
S100_DateTimeAttributeType [DateTime]	Combination of date and time representation as above.	The character 'T' must be used as the separator between the date and the time part.	20211223T170000
S100_URIAttributeType [URI]	A URI according to the RFC 3986.	URI must not contain white spaces; and must start with a 'scheme' followed by a ':'.	tel:+1-816-555-1212
S100_URLAttributeType [URL]	A URL according to the RFC 3986.		http://registry.iho.int
S100_URN_AttributeType [URN]	A URN as specified in RFC 2141	'scheme' = urn	urn:mrn:iala:aton:us:1234.5
S100_TruncatedDateAttributeType [S100_TruncatedDate]	A truncated version of a date.	CCYYMMDD All unspecified components are replaced by as many hyphens as the component has characters. Hence, the length of the encoding must be always 8 characters.	------01 ----1224
S100_CodelListAttributeType **[CharacterString]**	A code list value, represented as a CharacterString.	The URI of the code list dictionary is defined by the Feature Catalogue. See Parts 3 and 5 for more details.	eng

10a-5.1.5 Attribute field structure

Field Tag: **ATTR**	Field Name: Attribute

Subfield name	**Label**	**Format**	**Subfield content and specification**
Numeric attribute code	*NATC	b12	A valid attribute code as defined in the ATCS field of the Dataset General Information Record
Attribute index	ATIX	b12	Index (position) of the attribute in the sequence of attributes with the same code and the same parent (starting with 1)
Parent index	PAIX	b12	Index (position) of the parent complex attribute within this ATTR field (starting with 1). If the attribute has no parent (top level attribute) the value is 0
Attribute instruction	ATIN	b11	{1} – Insert {2} – Delete {3} – Modify
Attribute value	ATVL	A()	A string containing a valid value for the domain of the attribute specified by the subfields above

Data Descriptive Field

```
2600;&%/GAttribute▲*NATC!ATIX!PAIX!ATIN!ATVL▲(3b12,b11,A)▼
```

10a-5.2 Information Association field

10a-5.2.1 Encoding rules

An Information association is a link from one record to an information type record. An information type record can be referenced from any number of other records but at least one record should have an association to an information type record. Such associations will be encoded by means of the Information Association field (INAS). For each association a separate field has to be used. The association itself can have attributes. The attributes are encoded in the field by the same mechanism as described for the ATTR field. The same subfields are used at the end of the association field. Each association is uniquely addressed by the combination of the RRNM, RRID, IASS, and ROLE subfields.

The RRNM subfield is referencing the record name subfield (RCNM) and the RRID subfield is referencing the record id subfield (RCID) of the target record.

The Information Association Update Instruction INUI subfield is used to indicate if an association is to be inserted or deleted on update. For a base data set this field must have the value ‘Insert’.

10a-5.2.2 Information Association field structure

Field Tag: **INAS**	Field Name: Information Association

Subfield name	**Label**	**Format**	**Subfield content and specification**
Referenced Record name	RRNM	b11	Record name of the referenced record
Referenced Record identifier	RRID	b14	Record identifier of the referenced record
Numeric Information Association Code	NIAC	b12	A valid code for the information association as defined in the IACS field of the Dataset General Information Record
Numeric Association Role code	NARC	b12	A valid code for the role as defined in the ARCS field of the Dataset General Information Record
Information Association Update Instruction	IUIN	b11	{1} – Insert {2} – Delete {3} – Modify
Numeric attribute code	*NATC	b12	A valid attribute code as defined in the ATCS field of the Dataset General Information Record
Attribute index	ATIX	b12	Index (position) of the attribute in the sequence of attributes with the same code and the same parent (starting with 1)
Parent index	PAIX	b12	Index (position) of the parent complex attribute within this INAS field (starting with 1). If the attribute has no parent (top level attribute) the value is 0
Attribute Instruction	ATIN	b11	{1} – Insert {2} – Delete {3} – Modify
Attribute value	ATVL	A()	A string containing a valid value for the domain of the attribute specified by the subfields above

Data Descriptive Field

```
3600;&%/GInformation□Association▲RRNM!RRID!NIAC!NARC!IUIN\\*NATC!ATIX!PAIX
!ATIN!ATVL▲(b11,b14,2b12,b11,{3b12,b11,A})▼
```

10a-6 Data Set Descriptive records

10a-6.1 Data Set General Information record

10a-6.1.1 Encoding rules

This record encodes general information about the data set. This information includes identification, structural information and Metadata.

The Data Set Identification field contains information to identify the data set. This information is divided into three groups:

1) Information about the encoding;
2) Information about the data product;
3) Information about the data set itself.

The first group specifies the encoding specification on which the encoding is based and what version of that specification is applicable.

The second group defines the data product, the edition of the product specification and the profile used within the product. The product itself is specified by a unique identifier. Edition and Profile depend on the product specification and will be encoded as character strings.

The third group contains:

1) A file identifier of the data set;
2) A title of the data set;
3) The reference (issue) date of the data set;
4) The (default) language used in the data set;
5) An abstract about the data set;
6) The edition of the data set (may contain subversion/update number);
7) A list of topic categories according to ISO/IEC 19115-1 (see list):

Value of DSTC subfield	Topic Category
1	farming
2	biota
3	boundaries
4	climatologyMeterologyAtmosphere
5	economy
6	elevation
7	environment
8	geoscientificInformation
9	health
10	imageryBaseMapsEarthCover
11	intelligenceMilitary
12	inlandWaters
13	location
14	oceans
15	planningCadastre
16	society
17	structure
18	transportation
19	utilitiesCommunication
21	disaster

The Data Set Structure Information field contains some structural information. These are:

1) An origin offset used to shift the coordinate data being encoded such that higher precision can be carried in the region of the dataset.

2) The multiplication factors for the separate coordinate axes.

3) The number of the different kinds of records in the data file.

In an S-100 Feature catalogue all items are uniquely identifiable using the S100_FC_Item code which is a character string. This applies to Attributes, Information Types, Feature Types, Information Associations, Feature Associations and Association roles. In the interest of space and efficiency of the 8211 encoding it is desirable to use numeric identifiers for these items. To support this capability, the 8211 encoding includes a table for each item type that holds a listing of the S100_FC_Item codes used in the dataset. Each entry in the table carries the item code and the associated numeric code which will be used within the dataset everywhere that item type is referenced. These numeric codes are only guaranteed to be unique within one instance of a dataset. For example a Feature with code Coastline could be recorded in the Feature Type Codes field with a numeric code of 10. Then all the Coastline Features in the dataset would carry the numeric code of 10. In another dataset the numeric code for Coastline could be 15.

10a-6.1.2 Data Set General Information record structure

```
Data Set General Information record
   |
   |--DSID (13\\*1): Data Set Identification field
      |
      |--DSSI (13): Data Set Structure Information field
      |
      |-<0..1>-ATCS (*2): Attribute Codes field
      |
      |-<0..1>-ITCS (*2): Information Type Codes field
      |
      |-<0..1>-FTCS (*2): Feature Type Codes field
      |
      |-<0..1>-IACS (*2): Information Association Codes field
      |
      |-<0..1>-FACS (*2): Feature Association Codes field
      |
      |-<0..1>-ARCS (*2): Association Role Codes field
```

10a-6.1.2.1 Data Set Identification field structure

Field Tag: **DSID**	Field Name: Data Set Identification

Subfield name	Label	Format	Subfield content and specification
Record name	RCNM	b11	{10} - Data Set Identification
Record identification number	RCID	b14	Range: 1 to 2^{32}-2
Encoding specification	ENSP	A()	Encoding specification that defines the encoding
Encoding specification edition	ENED	A()	Edition of the encoding specification
Product identifier	PRSP	A()	Unique identifier for the data product as specified in the Product Specification
Product edition	PRED	A()	Edition of the Product Specification
Application profile	PROF	A()	Identifier that specifies a profile within the data product
Dataset file identifier	DSNM	A()	The file identifier of the dataset
Dataset title	DSTL	A()	The title of the dataset
Dataset reference date	DSRD	A(8)	The reference date of the dataset Format: YYYYMMDD according to ISO 8601
Dataset language	DSLG	A()	The (primary) language used in this dataset
Dataset abstract	DSAB	A()	The abstract of the dataset
Dataset edition	DSED	A()	The edition of the dataset
Dataset topic category	*DSTC	b11	A set of topic categories

Data Descriptive Field

```
3600;&%/GData□Set□Identification▲RCNM!RCID!ENSP!ENED!PRSP!PRED!PROF!DSNM!D
STL!DSRD!DSLG!DSAB!DSED\\*DSTC▲(b11,b14,7A,A(8),3A,{b11})▼
```

10a-6.1.2.2 Data Set Structure Information field structure

Field Tag: **DSSI**	Field Name: Data Set Structure Information

Subfield name	Label	Format	Subfield content and specification
Dataset Coordinate Origin X	DCOX	b48	Shift used to adjust x-coordinate before encoding
Dataset Coordinate Origin Y	DCOY	b48	Shift used to adjust y-coordinate before encoding
Dataset Coordinate Origin Z	DCOZ	b48	Shift used to adjust z-coordinate before encoding
Coordinate multiplication factor for x-coordinate	CMFX	b14	Floating point to integer multiplication factor for the x-coordinate or longitude
Coordinate multiplication factor for y-coordinate	CMFY	b14	Floating point to integer multiplication factor for the y-coordinate or latitude
Coordinate multiplication factor for z-coordinate	CMFZ	b14	Floating point to integer multiplication factor for the z-coordinate or depths or height
Number of Information Type records	NOIR	b14	Number of information records in the dataset
Number of Point records	NOPN	b14	Number of point records in the dataset
Number of Multi Point records	NOMN	b14	Number of multi point records in the dataset
Number of Curve records	NOCN	b14	Number of curve records in the dataset
Number of Composite Curve records	NOXN	b14	Number of composite curve records in the dataset
Number of Surface records	NOSN	b14	Number of surface records in the dataset
Number of Feature Type records	NOFR	b14	Number of feature records in the dataset

Data Descriptive Field

```
1600;&□□□Data□Set□Structure□Information▲DCOX!DCOY!DCOZ!CMFX!CMFY!CMFZ!NOIR
!NOPN!NOMN!NOCN!NOXN!NOSN!NOFR▲(3b48,10b14)▼
```

10a-6.1.2.3 Attribute Codes field structure

Field Tag: **ATCS**	Field Name: Attribute Codes

Subfield name	Label	Format	Subfield content and specification
Attribute Code	*ATCD	A	The code as defined in the Feature Catalogue
Attribute Numeric Code	ANCD	b12	The code used within the NATC subfield

Data Descriptive Field

```
2600;&□□□Attribute□Codes▲*ATCD!ANCD▲(A,b12)▼
```

10a-6.1.2.4 Information Type Codes field structure

Field Tag: **ITCS**	Field Name: Information Type Codes

Subfield name	Label	Format	Subfield content and specification
Information Type Code	*ITCD	A	The code as defined in the Feature Catalogue
Information Type Numeric Code	ITNC	b12	The code used within the NITC subfield

Data Descriptive Field

```
2600;&□□□Information□Type□Codes▲*ITCD!ITNC▲(A,b12)▼
```

10a-6.1.2.5 Feature Type Codes field structure

Field Tag: **FTCS**	Field Name: Feature Type Codes

Subfield name	Label	Format	Subfield content and specification
Feature Type Code	*FTCD	A	The code as defined in the Feature Catalogue
Feature Type Numeric Code	FTNC	b12	The code used within the NFTC subfield

Data Descriptive Field

```
2600;&□□□Feature□Type□Codes▲*FTCD!FTNC▲(A,b12)▼
```

10a-6.1.2.6 Information Association Codes field structure

Field Tag: **IACS**	Field Name: Information Association Codes

Subfield name	Label	Format	Subfield content and specification
Information Association Code	*IACD	A	The code as defined in the Feature Catalogue
Information Association Numeric Code	IANC	b12	The code used within the NIAC subfield

Data Descriptive Field

```
2600;&□□□Information□Association□Codes▲*IACD!IANC▲(A,b12)▼
```

10a-6.1.2.7 Feature Association Codes field structure

Field Tag: **FACS**	Field Name: Feature Association Codes

Subfield name	Label	Format	Subfield content and specification
Feature Association Code	*FACD	A	The code as defined in the Feature Catalogue
Feature Association Numeric Code	FANC	b12	The code used within the NFAC subfield

Data Descriptive Field

```
2600;&□□□Feature□Association□Codes▲*FACD!FANC▲(A,b12)▼
```

10a-6.1.2.8 Association Role Codes field structure

Field Tag: **ARCS**	Field Name: Association Role Codes

Subfield name	Label	Format	Subfield content and specification
Association Role Code	*ARCD	A	The code as defined in the Feature Catalogue
Association Role Numeric Code	ARNC	b12	The code used within the NARC subfield

Data Descriptive Field

```
2600;&□□□Association□Role□Codes▲*ARCD!ARNC▲(A,b12)▼
```

10a-6.2 Data Set Coordinate Reference System record

10a-6.2.1 Encoding rules

All two-dimensional coordinates in a dataset refer to one horizontal CRS. Three-dimensional coordinates refer to a compound CRS which consists of the horizontal CRS and a vertical CRS. There can be more than one vertical CRSs in a dataset one for each compound CRS.

The CRSH field contains the following information about the (single) CRS:

- The type of CRS (this implies the dimension of the coordinate system);
- The type of the associated coordinate system;
- The name of the CRS;
- An identifier in an external source (if the CRS is defined by referencing);
- An indication which external source is referenced;
- Information about this source (if it is not one from a predefined list).

If the CRS is not defined by referencing all details of the coordinate axes, the datum and if necessary about the used projection must be encoded. This has to done by means of the appropriate fields. In this case the CRSI subfield must be encoded empty and the CRSS subfield must have the value 255 (Not Applicable).

For more details on CRS refer to the Coordinate Reference System Component of this standard.

This encoding specification supports the following types of CRS's:

CRS Type	Dimension	CS Type	Axes	Type of Datum	CRST value	Remarks
2D Geographic	2	Ellipsoidal	Geodetic Latitude Geodetic Longitude	Geodetic	1	can be combined with a vertical CRS
3D Geographic	3	Ellipsoidal	Geodetic Latitude Geodetic Longitude Ellipsoidal Height	Geodetic	2	
Geocentric	3	Cartesian	Geocentric X Geocentric Y Geocentric Z	Geodetic	3	
Projected	2	Cartesian	Easting / Westing Northing / Southing	Geodetic	4	can be combined with a vertical CRS
Vertical	1	Vertical	Gravity Related Height or Gravity related Depth	Vertical	5	

The next table shows the supported coordinate axes:

Axis Type	Axis direction	AXTY value	Remarks
Geodetic Latitude	North	1	
Geodetic Longitude	East	2	
Ellipsoidal Height	Up	3	
Easting	East	4	
Northing	North	5	
Westing	West	6	
Southing	South	7	
Geocentric X	Geocentric X	8	
Geocentric Y	Geocentric Y	9	
Geocentric Z	Geocentric Z	10	
Gravity Related Height	Up	11	
Gravity Related Depth	Down	12	

This table shows the supported projections together with their set of parameters:

Name	PROM value	Parameter 1	Parameter 2	Parameter 3	Parameter 4	Parameter 5	EPSG code
Mercator	1	Latitude of 1st standard parallel [1])	Longitude of natural origin	-	-	-	9805
Transverse Mercator	2	Latitude of natural origin	Longitude of natural origin	Scale factor at natural origin	-	-	9807
Oblique Mercator	3	Latitude of projection centre	Longitude of projection centre	Azimuth of initial line	Angle from Rectified to Skew Grid	Scale factor on initial line	9815
Hotine Oblique Mercator	4	Latitude of projection centre	Longitude of projection centre	Azimuth of initial line	Angle from Rectified to Skew Grid	Scale factor on initial line	9812
Lambert Conic Conformal (1SP)	5	Latitude of natural origin	Longitude of natural origin	Scale factor at natural origin	-	-	9801
Lambert Conic Conformal (2SP)	6	Latitude of false origin	Longitude of false origin	Latitude of 1st standard parallel [2])	Latitude of 2nd standard parallel [3])	-	9802
Oblique Stereographic	7	Latitude of natural origin	Longitude of natural origin	Scale factor at natural origin	-	-	9809
Polar Stereographic	8	Latitude of natural origin [4])	Longitude of natural origin	Scale factor at natural origin	-	-	9810
Krovak Oblique Conic Conformal	9	Latitude of projection centre	Longitude of projection centre	Azimuth of initial line	Latitude of pseudo standard parallel	Scale factor on pseudo standard parallel	9819
American Polyconic	10	Latitude of natural origin	Longitude of natural origin	-	-	-	9818
Albers Equal Area	11	Latitude of false origin	Longitude of false origin	Latitude of 1st standard parallel [2])	Latitude of 2nd standard parallel [3])	-	9822
Lambert Azimuthal Equal Area	12	Latitude of natural origin	Longitude of natural origin	-	-	-	9820
New Zealand Mapgrid	13	Latitude of natural origin	Longitude of natural origin	-	-	-	9811

[1]) Latitude of true scale

[2]) Standard parallel nearer to equator

[3]) Standard parallel farther from equator

[4]) Must be either 90 degrees or -90 degrees

All latitudes and longitudes must be given in degrees (south and west are negative). Azimuths are given in degrees. For the detailed formulas of the projections refer to the EPSG documentation.

In case that both two-dimensional and three-dimensional coordinates are used in the same data set the three-dimensional coordinates must be described by a compound CRS. The two-dimensional coordinates refer to the first component (usually a 2D Geographic or Projected CRS).

Although all coordinates in a data set must refer to the same CRS different Vertical Datums can be used for the height or depth component of a coordinate tuple. Therefore the VDAT field can be repeated. For each Vertical Datum a unique identifier is defined. Those identifiers will be used in the 3D - coordinate fields to indicate which Vertical Datum is used. The encoding of the Coordinate Reference System record will be demonstrated with two examples. The first example specifies a compound CRS. The first component is a 2D Geographic CRS (WGS84) and the second component is a Vertical CRS for depth using the Vertical Datum: Mean Sea Level.

```
CSID: RCNM{15}!RCID{1}!NCRC{2}!
CRSH: CRIX{1}!CRST{1}!CSTY{1}!CRNM'WGS
      84'!CRSI'4326'!CRSS{2}!SCRI!
CRSH: CRIX{2}!CRST{5}!CSTY{3}!CRNM'Mean Sea Level Depth'!
      CRSI!CRSS{255}SCRI!
CSAX: AXTY{12}!AXUM{4}!
VDAT: DTNM'Mean Sea Level'!DTID'VERDAT3'!DTSR{2}!SCRI!
```

The second example encodes a projected CRS by defining the details.

```
CSID: RCNM{15}!RCID{1}!NCRS{1}!
CRSH: CRIX{1}!CRST{4}!CSTY{2}!CRNM'WGS84/UTM
      32N'!CRSI!CRSS{255}SCRI!
CSAX: AXTY{4}!AXUM{4}!AXTY{5}!AXUM{4}!
PROJ: PROM{2}!PRP1{0}!PRP2{9}!PRP3{0.9996}!PRP4{0}!PRP5{0}!
      FEAS{500000}!FNOR{0}!
GDAT: DTNM'World Geodetic System 1984'!ELNM'WGS 84'!ESMA{6378137}!
      ESPT{2}!ESPM{298.257223563}!CMNM'Greenwich'!CMGL{0}!
```

10a-6.2.2 Data Set Coordinate Reference System record structure

```
Data Set Coordinate Reference System record
   |
   |--CSID (3): Coordinate Reference System Record Identifier field
      |
      |-<1..*>-CRSH (7): Coordinate Reference System Header field
         |
         |-<0..1>-CSAX (*2): Coordinate System Axes field
         |
         |-<0..1>-PROJ (8): Projection field
         |
         *-<0..1>-GDAT (7): Geodetic Datum field
         |
         *-<0..1>-VDAT (4): Vertical Datum field
```

10a-6.2.2.1 Coordinate Reference System Record Identifier field structure

Field Tag: **CSID**	Field Name: Coordinate Reference System Record Identifier

Subfield name	**Label**	**Format**	**Subfield content and specification**
Record name	RCNM	b11	{15} - Coordinate Reference System Identifier
Record identification number	RCID	b14	Range: 1 to 2^{32}-2
Number of CRS Components	NCRC	b11	

Data Descriptive Field

```
1100;&□□□Coordinate□Reference□System□Record□Identifier▲RCNM!RCID!NCRC▲(b11
,b14,b11)▼
```

10a-6.2.2.2 Coordinate Reference System Header field structure

Field Tag: **CRSH**	Field Name: Coordinate Reference System Header

Subfield name	**Label**	**Format**	**Subfield content and specification**
CRS Index	CRIX	b11	Internal identifier of the CRS (Used for identifying the vertical CRS in C3DI or C3DF)

CRS Type	CRST	b11	see table
Coordinate System Type	CSTY	b11	{1} - Ellipsoidal CS {2} - Cartesian CS {3} - Vertical CS
CRS Name	CRNM	A()	Name of the Coordinate Reference System
CRS Identifier	CRSI	A()	Identifier of the CRS from an external source Empty if not defined by reference
CRS Source	CRSS	b11	{1} - IHO CRS Register {2} - EPSG {254} - Other Source {255} - Not Applicable
CRS Source Information	SCRI	A()	Information about the CRS source if CRSS = ‘Other Source’

Data Descriptive Field

```
1600;&%/GCoordinate□Reference□System□Header▲CRIX!CRST!CSTY!CRNM!CRSI!CRSS!
SCRI▲(3b11,2A,b11,A)▼
```

10a-6.2.2.3 Coordinate System Axes field structure

Field Tag: **CSAX**	Field Name: Coordinate System Axes

Subfield name	**Label**	**Format**	**Subfield content and specification**
Axis Type	*AXTY	b11	see table
Axis Unit of Measure	AXUM	b11	{1} - Degree {2} - Grad {3} - Radian {4} - Metre {5} - International foot {6} - US survey foot

Data Descriptive Field

```
2100;&□□□Coordinate□System□Axes▲*AXTY!AXUM▲(2b11)▼
```

10a-6.2.2.4 Projection field structure

Field Tag: **PROJ**	Field Name: Projection

Subfield name	**Label**	**Format**	**Subfield content and specification**
Projection Method	PROM	b11	see table
Projection Parameter 1	PRP1	b48	see table
Projection Parameter 2	PRP2	b48	see table
Projection Parameter 3	PRP3	b48	see table
Projection Parameter 4	PRP4	b48	see table
Projection Parameter 5	PRO5	b48	see table
False Easting	FEAS	b48	False easting (Units of measurement according to the coordinate axis ‘Easting’
False Northing	FNOR	b48	False northing (Units of measurement according to the coordinate axis ‘Northing’

Data Descriptive Field

```
1600;&□□□Projection▲PROM!PRP1!PRP2!PRP3!PRP4!PRP5!FEAS!FNOR!▲(b11,7b48)▼
```

10a-6.2.2.5 Geodetic Datum field structure

Field Tag: **GDAT**	Field Name: Geodetic Datum

Subfield name	Label	Format	Subfield content and specification
Datum Name	DTNM	A()	Name of the geodetic datum
Ellipsoid Name	ELNM	A()	Name of the ellipsoid
Ellipsoid semi major axis	ESMA	b48	Semi major axis of the ellipsoid in metre
Ellipsoid second parameter type	ESPT	b11	{1} - Semi minor axis in metres {2} - Inverse Flattening
Ellipsoid second parameter	ESPM	b48	The second defining parameter of the ellipsoid
Central Meridian Name	CMNM	A()	Name of the central meridian
Central Meridian Greenwich Longitude	CMGL	b48	Greenwich longitude of the central meridian in degrees

Data Descriptive Field

```
1600;&%/GGeodetic□Datum▲DTNM!ELNM!ESMA!ESPT!ESPM!CMNM!CMGL!▲(2A,b48,b11,b4
8,A,b48)▼
```

10a-6.2.2.6 Vertical Datum field structure

Field Tag: **VDAT**	Field Name: Vertical Datum

Subfield name	Label	Format	Subfield content and specification
Datum Name	DTNM	A()	Name of the vertical datum
Datum Identifier	DTID	A()	Identifier of the datum in an external source
Datum Source	DTSR	b11	{1} - IHO CRS Register {2} - Feature Catalogue {3} - EPSG {254} - Other Source {255} - Not Applicable
Datum Source Information	SCRI	A()	Information about the CRS source if DTSR = 'Other Source'

Data Descriptive Field

```
1600;&%/GVertical□Datum▲ DTNM!DTID!DTSR!SCRI▲( 2A,b11,A)▼
```

10a-7 Object Records

10a-7.1 Information Type record

10a-7.1.1 Encoding rules

Information types are pieces of information in a data set that can be shared between objects. They have attributes like feature types but are not related to any geometry. Information types may reference other information types. For this encoding it is important that an information type record must be stored prior to any record that references this record.

The object code must be a valid code in the feature catalogue that is defined for the data product. The record version will be initialized with 1 and will be incremented for any update of this record. The record update instruction indicates if an information type will be inserted, modified or deleted in an update. In a base data set the value will always be ‘Insert’.

10a-7.1.2 Information Type record structure

```
Information Type record
   |
   |--IRID (5): Information Type Record Identifier field
      |
      |-<0..*>-ATTR (*5): Attribute field
      |
      |-<0..*>-INAS (5\\*5): Information Association field
```

10a-7.1.2.1 Information Type Identifier field structure

Field Tag: **IRID**	Field Name: Information Type Record Identifier

Subfield name	Label	Format	Subfield content and specification
Record name	RCNM	b11	{150} – Information Type
Record identification number	RCID	b14	Range: 1 to 2^{32}-2
Numeric Information Type Code	NITC	b12	A valid information type code as defined in the ITCS field of the Dataset General Information Record
Record version	RVER	b12	RVER contains the serial number of the record edition
Record update instruction	RUIN	b11	{1} – Insert {2} – Delete {3} – Modify

Data Descriptive Field

```
1100;&□□□Information□Type□Record□Identifier▲RCNM!RCID!NITC!RVER!RUIN▲(b11,
b14,2b12,b11)▼
```

10a-7.2 Spatial type records

10a-7.2.1 Coordinate fields

10a-7.2.1.1 Encoding rules

Coordinates in a dataset are defined by the coordinate reference system (CRS). The CRS is defined in the Coordinate Reference System record. This record also defines the units of the coordinates.

The DSSI field of the Data Set General Information record can carry a local origin for the coordinates in a Data Set. When storing coordinates the Origin needs to be subtracted from the value, when reading coordinates from a dataset the Origin needs to be added back on to restore the CRS defined value.

Coordinates can be stored in two ways as floating point numbers or as integer numbers. In the latter case the stored integer value is calculated by the multiplication of the real coordinate and a multiplication factor. Those factors are defined for each coordinate axis in the DSSI field of the Data Set General Information record. With these factors the stored value can be transformed into the real coordinate according to the coordinate reference system (CRS).

The coordinates are transformed as follows:

```
x = DCOX + XCOO / CMFX
y = DCOY + YCOO / CMFY
z = DCOZ + ZCOO / CMFZ
```

Note that the values of (CMFX, CMFY and CMFZ) should be set to 1 if the coordinates are stored as floating point values.

If the coordinate field allows more than one coordinate tuple the update must maintain the order of the coordinates. Each update of a coordinate stream is therefore defined by an index into the coordinate field(s) of the target record, an update instruction and the number of coordinates in the coordinate field(s) of the update record.

Note that the index and the number refer to coordinate tuples, not to single coordinates. The index will start with 1.

10a-7.2.1.2 Coordinate Control field structure

Field Tag: **COCC** [Upd]	Field Name: Coordinate Control

Subfield name	Label	Format	Subfield content and specification
Coordinate Update Instruction	COUI	b11	{1} – Insert {2} – Delete {3} – Modify
Coordinate Index	COIX	b12	Index (position) of the addressed coordinate tuple within the coordinate field(s) of the target record
Number of Coordinates	NCOR	b12	Number of coordinate tuples in the coordinate field(s) of the update record

Data Descriptive Field

```
1100;&□□□Coordinate□Control▲COUI!COIX!NCOR▲(b11,2b12)▼
```

10a-7.2.1.3 2-D Integer Coordinate Tuple field structure

Field Tag: **C2IT**	Field Name 2-D Integer Coordinate Tuple

Subfield name	Label	Format	Subfield content and specification
Coordinate in Y axis	YCOO	b24	Y-coordinate or latitude
Coordinate in X axis	XCOO	b24	X-coordinate or longitude

Data Descriptive Field

```
1100;&□□□2-D□Integer□Coordinate□Tuple▲YCOO!XCOO▲(2b24)▼
```

10a-7.2.1.4 3-D Integer Coordinate Tuple field structure

Field Tag: **C3IT**	Field Name: 3-DInteger Coordinate Tuple

Subfield name	Label	Format	Subfield content and specification
Vertical CRS Id	VCID	b11	Internal identifier of the Vertical CRS
Coordinate in Y axis	YCOO	b24	Y-coordinate or latitude
Coordinate in X axis	XCOO	b24	X-coordinate or longitude
Coordinate in Z axis	ZCOO	b24	Z-coordinate (depth or height)

Data Descriptive Field

```
1100;&□□□3-D□Integer□Coordinate□Tuple▲VCID!YCOO!XCOO!ZCOO▲(b11,3b24)▼
```

10a-7.2.1.5 2-D Floating Point Coordinate Tuple field structure

Field Tag: **C2FT**	Field Name 2-D Floating Point Coordinate Tuple

Subfield name	**Label**	**Format**	**Subfield content and specification**
Coordinate in Y axis	YCOO	b48	Y-coordinate or latitude
Coordinate in X axis	XCOO	b48	X-coordinate or longitude

Data Descriptive Field

```
2200;&□□□2-D□Floating□Point□Coordinate□Tuple▲YCOO!XCOO▲(2b48)▼
```

10a-7.2.1.6 3-D Floating Point Coordinate Tuple field structure

Field Tag: **C3FT**	Field Name: 3-D Floating Point Coordinate Tuple

Subfield name	**Label**	**Format**	**Subfield content and specification**
Vertical CRS Id	VCID	b11	Internal identifier of the Vertical CRS
Coordinate in Y axis	YCOO	b48	Y-coordinate or latitude
Coordinate in X axis	XCOO	b48	X-coordinate or longitude
Coordinate in Z axis	ZCOO	b48	Z-coordinate (depth or height)

Data Descriptive Field

```
3600;&□□□3-D□Floating□Point□Coordinate□Tuple▲VCID!YCOO!XCOO!ZCOO▲(b11,3b48)▼
```

10a-7.2.1.7 2-D Integer Coordinate List field structure

Field Tag: **C2IL**	Field Name 2-D Integer Coordinate List

Subfield name	**Label**	**Format**	**Subfield content and specification**
Coordinate in Y axis	*YCOO	b24	Y-coordinate or latitude
Coordinate in X axis	XCOO	b24	X-coordinate or longitude

Data Descriptive Field

```
2100;&□□□2-D□Integer□Coordinate□List▲*YCOO!XCOO▲(2b24)▼
```

10a-7.2.1.8 3-D Integer Coordinate List field structure

Field Tag: **C3IL**	Field Name: 3-DInteger Coordinate List

Subfield name	**Label**	**Format**	**Subfield content and specification**
Vertical CRS Id	VCID	b11	Internal identifier of the Vertical CRS
Coordinate in Y axis	*YCOO	b24	Y-coordinate or latitude
Coordinate in X axis	XCOO	b24	X-coordinate or longitude
Coordinate in Z axis	ZCOO	b24	Z-coordinate (depth or height)

Data Descriptive Field

```
3100;&□□□3-D□Integer□Coordinate□List▲VCID\\*YCOO!XCOO!ZCOO▲(b11,{3b24})▼
```

10a-7.2.1.9 2-D Floating Point Coordinate List field structure

Field Tag: **C2FL**	Field Name 2-D Floating Point Coordinate List

Subfield name	Label	Format	Subfield content and specification
Coordinate in Y axis	*YCOO	b48	Y-coordinate or latitude
Coordinate in X axis	XCOO	b48	X-coordinate or longitude

Data Descriptive Field

```
2200;&□□□2-D□Floating□Point□Coordinate□List▲*YCOO!XCOO▲(2b48)▼
```

10a-7.2.1.10 3-D Floating Point Coordinate List field structure

Field Tag: **C3FL**	Field Name: 3-D Floating Point Coordinate List

Subfield name	Label	Format	Subfield content and specification
Vertical CRS Id	VCID	b11	Internal identifier of the Vertical CRS
Coordinate in Y axis	*YCOO	b48	Y-coordinate or latitude
Coordinate in X axis	XCOO	b48	X-coordinate or longitude
Coordinate in Z axis	ZCOO	b48	Z-coordinate (depth or height)

Data Descriptive Field

```
3600;&□□□3-D□Floating□Coordinate□List▲VCID\\*YCOO!XCOO!ZCOO▲(b11,{3b24})▼
```

10a-7.2.1.11 Knots

Knots are parameters used in spline curves to control the shape of the curve. Each knot defines a value in parameter space of the spline, which will be used to define the spline basis functions. The knot data type holds information on knot multiplicity. The parameter values in the knot array must be monotonic and strictly increasing; that is, each value must be greater than its predecessor.

Field Tag: **KNOT**	Field Name Knots

Subfield name	Label	Format	Subfield content and specification
Knot multiplicity	*KMUL	b11	The multiplicity of the knot
Knot value	KVAL	b48	The value of the knot

Data Descriptive Field

```
1600;&□□□Knot▲KMUL!KVAL▲(b11,b48)▼
```

10a-7.2.1.12 Derivatives

The derivatives field encodes the derivatives of a curve at a point. Any missing values must be encoded as ‘omitted’ values (see clause 10a-3.5). Derivatives are encoded in order beginning with the first-order derivative.

The derivatives are given in terms of their X and Y components. In this edition of S-100 derivatives are defined only in 2-D because splines are only in 2-D.

The derivatives are defined in floating-point and integer formats to be used with the corresponding types of coordinate fields.

Field Tag: **DRVF**	Field Name 2-D Derivative List Float

Subfield name	Label	Format	Subfield content and specification
Y component of point at which defined	YCOO	b48	The Y component of the point at which the derivatives are defined
X component of point at which defined	XCOO	b48	The X component of the point at which the derivatives are defined
Highest order of derivative	DRVO	b11	The highest order derivative in the list
Y offset	*YDRV	b48	The Y component of the n'th derivative.
X offset	XDRV	b48	The X component of the n'th derivative.

Data Descriptive Field

```
3600;&□□□2-D□Derivative□List□Float▲YCOO!XCOO!DRVO!\\*YDRV!XDRV▲(2b48,b11,2
b48)▼
```

Field Tag: **DRVI**	Field Name 2-D Derivative List Integer

Subfield name	Label	Format	Subfield content and specification
Y component of point at which defined	YCOO	b24	The Y coordinate of the point at which the derivatives are defined
X component of point at which defined	XCOO	b24	The X coordinate of the point at which the derivatives are defined
Highest order of derivative	DRVO	b11	The highest order derivative in the list
Y component of derivative	*YDRV	b24	The Y component of the n'th derivative.
X component of derivative	XDRV	b24	The X component of the n'th derivative.

Data Descriptive Field

```
3600;&□□□2-D□Derivative□List□Integer▲YCOO!XCOO!DRVO!\\*YDRV!XDRV▲(2b24,b11
,2b24)▼
```

10a-7.2.2 Point record

10a-7.2.2.1 Encoding rules

A point is a zero-dimensional spatial object. It will be encoded with the Point record. This record contains the Point Record Identifier field. With the RCNM and RCID subfields every point must be uniquely identifiable within a data set. A point can have attributes and associations to information types.

Each point has exactly one coordinate field with exactly one coordinate tuple. Points can have both 2D or 3D coordinates.

Since there is only one coordinate tuple no special mechanism is necessary to address a coordinate for updating. When the coordinate of a point is to be updated the update record will contain a coordinate field with the new coordinate. The dimension of the coordinate in the update record must be the same as in the target record.

10a-7.2.2.2 Point record structure

```
Point record
   |
   |--PRID (4): Point Record Identifier field
      |
      |-<0..*>-INAS (5\\*5): Information Association field
      |
```

```
|alternate coordinate representations
|
*--C2IT (2): 2-D Integer Coordinate Tuple field
|
*--C3IT (4): 3-D Integer Coordinate Tuple field
|
*--C2FT (2): 2-D Floating Point Coordinate Tuple field
|
*--C3FT (4): 3-D Floating Point Coordinate Tuple field
```

10a-7.2.2.2.1 Point Record Identifier field structure

Field Tag: **PRID**	Field Name: Point Record Identifier

Subfield name	Label	Format	Subfield content and specification
Record name	RCNM	b11	{110} – Point
Record identification number	RCID	b14	Range: 1 to 2^{32}-2
Record version	RVER	b12	RVER contains the serial number of the record edition
Record update instruction	RUIN	b11	{1} – Insert {2} – Delete {3} – Modify

Data Descriptive Field

```
1100;&□□□Point□Record□Identifier▲RCNM!RCID!RVER!RUIN▲(b11,b14,b12,b11)▼
```

10a-7.2.3 Multi Point record

10a-7.2.3.1 Encoding rules

A Multi Point is an aggregation of zero-dimensional spatial objects. It will be encoded with the Multi Point record. Each Multi Point must have a unique identifier (RCNM + RCID) stored in the Multi Point Record Identifier field. Like any other spatial object Multi Points can have attributes and associations to information types.

Coordinates will be stored by one type of the coordinate list fields. The field can be repeated and in one field can be multiple coordinate tuples. If multiple coordinate list fields are used they must be all of the same type. If 3D-coordinates are used for the Multi Point they must all refer to the same Vertical Datum.

On updating the Coordinate control field defines which coordinates in the target record will be updated. Three kinds of updates are possible as defined by the Coordinate Update Instruction subfield (COUI):

1) Insert
   Coordinates encoded in the coordinate field(s) of the update record must be inserted in the coordinate field(s) of the target record. The Coordinate Index subfield (COIX) indicates the index where the new coordinates are to be inserted. The first coordinate has the index 1. The number of coordinates to be inserted is given in the Number of Coordinates subfield (NCOR).
2) Delete
   Coordinates must be deleted from the coordinate field(s) of the target record. The deletion must start at the index specified in the COIX subfield. The number of coordinates to be removed is given in the NCOR subfield.
3) Modify
   Coordinates encoded in the coordinate field(s) of the update record must replace the addressed coordinate(s) in the coordinate field(s) of the target record. The replacement must start at the index given in the COIX subfield. The number of coordinates to be replaced is given in the NCOR subfield.

Note that the index and number as given in the COIX and NCOR subfields are regarded to coordinate tuples not to single coordinates.

If several operations are necessary to update the coordinates of one target record each operation shall be encoded in a separate update record. Note that indices always refer to the latest version of the record; that is if the indices of coordinates have changed by one update record these changes have to be taken into account in every subsequent update record.

All coordinates in an update record must be stored in the same type of Coordinate field that is used in the target record and for 3D-coordinates the must refer to the same Vertical Datum as the coordinates in the target record.

10a-7.2.3.2 Multi Point record structure

```
Multi Point record
   |
   |--MRID (4): Multi Point Record Identifier field
      |
      |-<0..*>-INAS (5\\*5): Information Association field
      |
      |-<0..1>-COCC (3): Coordinate Control field
      |
      |alternate coordinate representations
      |
      *-<0..*>-C2IL (*2): 2-D Integer Coordinate List field
      |
      *-<0..*>-C3IL (1\\*3): 3-D Integer Coordinate List field
      |
      *-<0..*>-C2FL (*2): 2-D Floating Point Coordinate List field
      |
      *-<0..*>-C3FL (1\\*3): 3-D Floating Point Coordinate List field
```

10a-7.2.3.2.1 Multi Point Record Identifier field structure

Field Tag: **MRID**	Field Name: Multi Point Record Identifier

Subfield name	Label	Format	Subfield content and specification
Record name	RCNM	b11	{115} – Multi Point
Record identification number	RCID	b14	Range: 1 to 2^{32}-2
Record version	RVER	b12	RVER contains the serial number of the record edition
Record update instruction	RUIN	b11	{1} – Insert {2} – Delete {3} – Modify

Data Descriptive Field

```
1100;&□□□Multi□Point□Record□Identifier▲RCNM!RCID!RVER!RUIN▲(b11,b14,b12,b1
1)▼
```

10a-7.2.4 Curve record

10a-7.2.4.1 Encoding rules

A Curve is a one-dimensional spatial object. It consists of one or more segments which define the geometry of the curve. All segments of one curve define one contiguous path. The geometry of a segment is given by a set of control points (coordinates) and an interpolation method. As with any other spatial object, curves can have associations to information types. A curve can have associations to points which define the topological boundaries (the ends) of the curve. Those points must be coincident with the start of the first segment or with the end of the latest segment respectively. The association with such points will be encoded by means of the Point Association field (PTAS).

For each segment, one Segment Header field (SEGH) has to be encoded followed by the Coordinate Control field (update records only) and Coordinate fields.

- For segments with the INTP subfield set to 7 (CircularArcCenterPointWithRadius) a parameter field (CIPM or ARPM) must follow the Coordinate field to define the additional parameter of such segments. The CIPM (Circle Parameter field) must be used if the segment is a full circle and the ARPM (Arc Parameter field) must be used for circular arcs. Note that for such segments there is exactly one control point.
- For segments with the INTP field set to 8 (polynomialSpline) or 9 (bezierSpline), the polynomial spline parameter field (PSPL) must follow the Coordinate field to define the additional parameter for such segments. The Knot fields are required only if the knots are not uniform (knotSpec is other than 1).
- For segments with the INTP field set to 10 (bSpline), the spline parameter field (SPLI) must follow the Coordinate field to define the additional parameters for spline segments. The Knot fields are required only if the knots are not uniform (knotSpec is other than 1).
- For segments with the INTP field set to 11 (blendedParabolic) no additional parameters are needed. The control (data) points given in the Coordinate fields and the interpolation type suffice to define the curve segment. Note that for closed segments the start and end points of the segment must overlap in order to produce a smooth closed curve (see Part 7 clause 7-4.2.2.2).

Coordinates of control points can be stored in the following fields: C2IL, C2FL, C3IL, or C3FL. Those fields, coordinate list fields, can be repeated and can carry multiple coordinate tuples (exept for INTP equal to 7 see above).

If multiple coordinate list fields are used they must be all of the same type. If 3D-coordinates are used for the segment they must all refer to the same Vertical Datum.

For the Point Association field no special update instruction is needed. The association defined in the update record will replace the respective association in the target record.

For segments the order is important and must be maintained during the update. Therefore a special control field for segments will be used during update. The order of segments in a curve is defined by the sequence of Segment Header fields in the record. To update this sequence the Segment Control field (SECC) is used.

Three instructions can be defined in the SEUI subfield:

1) Insert
   Segments of the update record has to be inserted into the target record. The SEIX subfield specifies the index (position) where the segments are to be inserted. The subfield NSEG subfield gives the number of segments to be inserted.
2) Delete
   Segments must be deleted from the target record. The subfields SEIX and NSEG specify where and how many segments are to be deleted.
3) Modify
   Segments of the target record must be modified according to the encoded instructions in the update record. Each segment that is to be modified must have at a Segment Header filed, a Coordinate Control field and if necessary the appropriate Coordinate fields. The SEIX subfield indicates the first segment to be modified and the NSEG subfield gives the number of segments to be modified. All segments to be modified with one update record must be contiguous in the target record. Otherwise more than one update record has to be used.

When the coordinates of the control points of a segment are to be modified, this has to be done by means of the Coordinate Control field. It defines which coordinates in the target record will be updated. Three kinds of updates are possible and are defined by the Coordinate Update Instruction subfield (COUI):

1) Insert
   Coordinates encoded at the coordinate field(s) of the update records segment must be inserted in the coordinate field(s) of the corresponding target records segment. The Coordinate Index subfield (COIX) indicates the index where the new coordinates are inserted. The first coordinate has the index 1. The number of coordinates to be inserted is given in the Number of Coordinates subfield (NCOR).
2) Delete
   Coordinates must be deleted from the coordinate field(s) of the corresponding target records segment. The deletion must start at the index specified in the COIX subfield. The number of

coordinates to be removed is given in the NCOR subfield.

3) Modify
Coordinates encoded in the coordinate field(s) of the update records segment must be replace the addressed coordinate(s) in the coordinate field(s) of the corresponding target records segment. The replacement must start at the index given in the COIX subfield. The number of coordinates to be replaced is given in the NCOR subfield.

Note that the index and number as given in the COIX and NCOR subfields refer to coordinate tuples not to single coordinates.

All coordinates in an update record must be stored in the same type of Coordinate field that is used in the target record and for 3D-coordinates the must refer to the same Vertical Datum as the coordinates in the target record.

10a-7.2.4.2 Curve record structure

```
Curve record
   |
   |--CRID (4): Curve Record Identifier field
     |
     |-<0..*>-INAS (5\\*5): Information Association field
     |
     |-<0..1>-PTAS (*3): Point Association field
     |
     |-<0..1>-SECC (3): Segment Control field
     |
     |-<0..*>-SEGH (1): Segment Header field
       |
       |-<0..1>-COCC (3): Coordinate Control Field
       |
       |alternate coordinate representations
       |
       *-<0..*>-C2IL (*2): 2-D Integer Coordinate List field
       |
       *-<0..*>-C3IL (1\\*3): 3-D Integer Coordinate List field
       |
       *-<0..*>-C2FL (*2): 2-D Floating Point Coordinate List field
       |
       *-<0..*>-C3FL (1\\*3): 3-D Floating Point Coordinate List
     |
       |alternate parameter for circle and arc segments
       |
       *-<0..1>-CIPM (6): Circle Parameter field
       |
       *-<0..1>-ARPM (6): Arc Parameter field
       |
       |alternate parameters for spline segments
       |
       *-<0..1>-SPLI (1): Spline Parameter field
       |  |
       |  *-<0..1>-KNOT (*2) Knots array field
       |
       *-<0..1>-PSPL (1): Polynomial Spline Parameter field
         |
         *-<0..1>-KNOT (*2) Knots array field
         |
         |alternate coordinate representations
         |
         *-<0..1>-DRVF (*4) Derivatives field (floating point)
         |
         *-<0..1>-DRVI (*4) Derivatives field (Integer)
```

10a-7.2.4.2.1 Curve Record Identifier field structure

Field Tag: **CRID**	Field Name: Curve Record Identifier

Subfield name	Label	Format	Subfield content and specification
Record name	RCNM	b11	{120} – Curve
Record identification number	RCID	b14	Range: 1 to 2^{32}-2
Record version	RVER	b12	RVER contains the serial number of the record edition
Record update instruction	RUIN	b11	{1} – Insert {2} – Delete {3} – Modify

Data Descriptive Field

```
1100;&□□□Curve□Record□Identifier▲RCNM!RCID!RVER!RUIN▲(b11,b14,b12,b11)▼
```

10a-7.2.4.2.2 Point Association field structure

Field Tag: **PTAS**	Field Name: Point Association

Subfield name	Label	Format	Subfield content and specification
Referenced Record name	*RRNM	b11	Record name of the referenced record
Referenced Record identifier	RRID	b14	Record identifier of the referenced record
Topology indicator	TOPI	b11	{1} – Beginning point {2} – End point {3} – Beginning & End point

Data Descriptive Field

```
2100;&□□□Point□Association▲*RRNM!RRID!TOPI▲(b11,b14,b11)▼
```

10a-7.2.4.2.3 Segment Control field structure

Field Tag: **SECC** [Upd]]	Field Name: Segment Control

Subfield name	Label	Format	Subfield content and specification
Segment update instruction	SEUI	b11	{1} – Insert {2} – Delete {3} – Modify
Segment index	SEIX	b12	Index (position) of the addressed segment in the target record
Number of segments	NSEG	b12	Number of segments in the update record

Data Descriptive Field

```
1100;&□□□Segment□Control▲SEUI!SEIX!NSEG▲(b11,2b12)▼
```

10a-7.2.4.2.4 Segment Header field structure

Field Tag: **SEGH**	Field Name: Segment Header

Subfield name	Label	Format	Subfield content and specification
Interpolation	INTP	b11	{1} – Linear {2} – Arc3Points {3} – Geodesic {4} – Loxodromic {5} – Elliptical {6} – Conic {7} – CircularArcCenterPointWithRadius {8} – polynomialSpline {9} – bezierSpline {10} – bSpline {11} – blendedParabolic

Data Descriptive Field

```
1100;&□□□Segment□Header▲INTP ▲(b11)▼
```

10a-7.2.4.2.5 Circle Parameter field structure

Field Tag: **CIPM**	Field Name: Circle Parameter

Subfield name	Label	Format	Subfield content and specification
Radius	RADI	b48	Radius of the circle
Unit of Radius	RADU	b11	{1} – Metres {2} – Yards {3} – Kilometres {4} – Statute miles {5} – Nautical miles

Data Descriptive Field

```
1100;&□□□Circle□Parameter▲RADI!RADU▲(b48,b11)▼
```

10a-7.2.4.2.6 Arc Parameter field structure

Field Tag: **ARPM**	Field Name: Arc Parameter

Subfield name	Label	Format	Subfield content and specification
Radius	RADI	b48	Radius of the circle
Unit of Radius	RADU	b11	{1} – Metres {2} – Yards {3} – Kilometres {4} – Statute miles {5} – Nautical miles
Start Bearing Angle	SBRG	b48	In decimal degrees, range [0.0, 360.0]
Angular distance	ANGL	b48	In decimal degrees [-360.0, 360.0

Data Descriptive Field

```
1100;&□□□Arc□Parameter▲RADI!RADU!SBRG!ANGL▲(b48,b11,2b48)▼
```

10a-7.2.4.2.7 Spline Parameter field structure

Field Tag: **SPLI**	Field Name: Spline Parameter

Subfield name	Label	Format	Subfield content and specification
Degree	DEGR	b11	The degree of the interpolating polynomial.
KnotSpec	KSPC	b11	{1} – uniform {2} – quasiUniform {3} – piecewiseBezier {4} – nonUniform
Is Rational	RTNL	b11	{1} – the spline is a rational spline {2} – the spline is not a rational spline

Data Descriptive Field

```
1100;&□□□Spline□Parameter▲DEGR!KSPC!RTNL▲(3b11)▼
```

10a-7.2.4.2.8 Polynomial Spline Parameter field structure

Field Tag: **PSPL**	Field Name: Polynomial Spline Parameter

Subfield name	Label	Format	Subfield content and specification
Degree	DEGR	b11	Radius of the circle
KnotSpec	KSPC	b11	{1} – uniform {2} – quasiUniform {3} – piecewiseBezier {4} – nonUniform
Is Rational	RTNL	b11	{1} – the spline is a rational spline {2} – the spline is not a rational spline
Number of derivatives at start and end	NDRV	b11	The number or derivatives at each end. The number of derivatives at the start and end must be the same. If the start and end have different numbers of derivatives the missing values must be encoded as ‘omitted’ values (see 10a-3.5)
Number derivatives Interior	NDVI	b11	The number of interior derivatives required to be continuous. E.g., “2” means the first and second interior derivatives must be continuous

Data Descriptive Field

```
1100;&□□□Polynomial□Spline□Parameter▲DEGR!KNUM!KSPC!RTNL!NDRV!NDVI▲(5b11)▼
```

10a-7.2.5 Composite Curve record

10a-7.2.5.1 Encoding rules

Composite Curves are one-dimensional spatial objects that are composed of other curves. A composite curve itself is a contiguous path; that is, the end of one component must be coincident with the start of the next component. Components are curves, although the direction in which they are used may be opposite to the direction in which the curve is defined originally. Which direction is used will be encoded in the ORNT subfield of the Curve Component field (CUCO).

The topological boundaries are not encoded explicitly. The beginning node is taken from the first component and the end node is taken from the last component. Which boundary is taken depends on the ORNT subfield.

Attributes and associations to information types can be encoded as for all other spatial objects.

Composite curves can have other composite curves as components. In this case the record of the component must be stored prior to the record which references the component.

Since the order of components is essential for the definition of the composite curve it must be maintained during an update. Therefore, a special control field is used to update the sequence of components. This field contains an update instruction subfield (CCUI) that can have three values:

1) Insert
   The components of the update record must be inserted in the sequence of components defined in the target record. The CCIX will define the index (position) where the components are to be inserted. The first component has the index 1. The NCCO subfield gives the number of components in the update record. The new components must be added to the dataset before references to them can be inserted into the composite curve.
2) Delete
   Components must be deleted from the target record. The CCIX subfield will specify the index (position) of the first components to be deleted, The NCCO subfield gives the number of components to be deleted. Note that the component is only deleted from the sequence of components of the composite curve not from the data set.
3) Modify
   The components in the target record will be replaced by the components in the update record. The first component to be replaced is given by the subfield CCIX, the number of components to be replaced is specified by the subfield NCCO. New components must be added to the dataset before references to them can be applied to the composite curve.

If more than one instruction is necessary to update the sequence of components multiple update records have to be encoded. Note that indices always refer to the latest version of the record, that is if the indices of components have changed by one update record these changes have to be taken into account in every subsequent update record.

10a-7.2.5.2 Composite Curve record structure

```
Composite Curve record
   |
   |--CCID (4): Composite Curve Record Identifier field
      |
      |-<0..*>-INAS (5\\*5): Information Association field
      |
      |-<0..1>-CCOC (3): Curve Component Control field
      |
      |-<0..*>-CUCO (*3): Curve Component field
```

10a-7.2.5.2.1 Composite Curve Record Identifier field structure

Field Tag: **CCID**	Field Name: Composite Curve Record Identifier

Subfield name	Label	Format	Subfield content and specification
Record name	RCNM	b11	{125} – Composite Curve
Record identification number	RCID	b14	Range: 1 to 2^{32}-2
Record version	RVER	b12	RVER contains the serial number of the record edition
Record update instruction	RUIN	b11	{1} – Insert {2} – Delete {3} – Modify

Data Descriptive Field

```
1100;&□□□Composite□Curve□Record□Identifier▲RCNM!RCID!RVER!RUIN▲(b11,b14,b1
2,b11)▼
```

10a-7.2.5.2.2 Curve Component Control field structure

Field Tag: **CCOC** [Upd]	Field Name: Curve Component Control

Subfield name	Label	Format	Subfield content and specification
Curve Component update instruction	CCUI	b11	{1} – Insert {2} – Delete {3} – Modify
Curve Component index	CCIX	b12	Index (position) of the addressed Curve record pointer within the CUCO field(s) of the target record
Number of Curve Components	NCCO	b12	Number of Curve record pointer in the CUCO field(s) of the update record

Data Descriptive Field

```
1100;&□□□Curve□Component□Control▲CCUI!CCIX!NCCO▲(b11,2b12)▼
```

10a-7.2.5.2.3 Curve Component field structure

Field Tag: **CUCO**	Field Name: Curve Component

Subfield name	Label	Format	Subfield content and specification
Referenced Record name	*RRNM	b11	Record name of the referenced record
Referenced Record identifier	RRID	b14	Record identifier of the referenced record
Orientation	ORNT	b11	{1} – Forward {2} – Reverse

Data Descriptive Field

```
2100;&□□□Curve□Component▲*RRNM!RRID!ORNT▲(b11,b14,b11)▼
```

10a-7.2.6 Surface record

10a-7.2.6.1 Encoding rules

A surface is a two-dimensional spatial object. It is defined by its boundaries. Each boundary is a closed curve. Closed means that the start and the end point of that curve are coincident. A surface has exactly one exterior boundary and can have zero or more interior boundaries (holes in the surface).

All interior boundaries must be completely inside the exterior boundary and no interior boundary must be inside another interior boundary. Boundaries must not intersect but a tangential touch is allowed. Those boundaries, also called rings, are encoded with the Ring Association field. Each ring will be encoded by a reference to a curve record (RRNM and RRID), the orientation (ORNT) in which the curve is used and the indication whether this ring is exterior or interior (USAG). In Addition each ring is encoded with an update instruction (RAUI). Since the order how the ring associations are encoded is arbitrary there is no special update field to add or remove rings from a surface definition. This will be made with the Ring Association field and the appropriate Ring Association Update Instruction (RAUI) subfield.

10a-7.2.6.2 Surface record structure

```
Surface record
   |
   |--SRID (4): Surface Record Identifier field
      |
      |-<0..*>-INAS (5\\*5): Information Association field
      |
      |-<1..*>-RIAS (*5): Ring Association field
```

10a-7.2.6.2.1 Surface Record Identifier field structure

Field Tag: **SRID**	Field Name: Surface Record Identifier

Subfield name	Label	Format	Subfield content and specification
Record name	RCNM	b11	{130} – Surface
Record identification number	RCID	b14	Range: 1 to 2^{32}-2
Record version	RVER	b12	RVER contains the serial number of the record edition
Record update instruction	RUIN	b11	{1} – Insert {2} – Delete {3} – Modify

Data Descriptive Field

```
1100;&□□□Surface□Record□Identifier▲RCNM!RCID!RVER!RUIN▲(b11,b14,b12,b11)▼
```

10a-7.2.6.2.2 Ring Association field structure

Field Tag: **RIAS**	Field Name: Ring Association

Subfield name	Label	Format	Subfield content and specification
Referenced Record name	*RRNM	b11	Record name of the referenced record
Referenced Record identifier	RRID	b14	Record identifier of the referenced record
Orientation	ORNT	b11	{1} – Forward {2} – Reverse
Usage indicator	USAG	b11	{1} – Exterior {2} – Interior
Ring Association update instruction	RAUI	b11	{1} – Insert {2} – Delete

Data Descriptive Field

```
2100;&□□□Ring□Association▲*RRNM!RRID!ORNT!USAG!RAUI▲(b11,b14,3b11)▼
```

10a-7.3 Feature Type record

10a-7.3.1 Encoding rules

An instance of a feature type is implemented in the data structure as a feature record. Feature types are listed in the feature catalogue of the data product. For each feature type the feature catalogue defines permissible attributes and associations. The feature catalogue defines also the two roles for each feature to feature association.

An S-100 compliant feature catalogue identifies 4 categories of feature types:

1) Meta feature;
2) Cartographic feature;
3) Geographic feature;
4) Theme feature.

Each category is implemented in the structure as a feature record and encoded in the same manner.

In the FRID field the code of the feature type is encoded. It must be a valid type from the feature catalogue of the data product. Note that for products using this encoding the feature catalogue must provide a 16-bit integer code.

The FOID field encodes a unique identifier for the instance of a feature type. Instances that are split into separate parts can have the same Feature Object Identifier indicating that this is the same feature object. This is possible for parts in the same data set but also for feature objects in different data sets. The latter case allows to identify parts of the same feature object in adjacent data sets or to determine identical feature objects in different scale bands.

The Feature Object Identifier is only used for implicit relationships not for referencing records directly. That is always done by the combination of the Referenced Record Name (RRNM) and Referenced Record Identifier (RRID).

Feature types are characterised by attributes and can have additional information associated by means of information types. Attributes are encoded by the Attribute field (ATTR) whereas the Information Association field is used for encoding the associations to information types.

The location of a feature object is defined by spatial objects. The association to these spatial objects is encoded with the Spatial Association field. It consists of a reference to the spatial object, an orientation flag, and two values which specifies the scale range for depicting the feature with the referenced geometry. The orientation flag is only necessary if the direction (of a curve) is meaningful for the feature object (for example a one-way street).

Feature types can have associations to other feature types. These associations including their roles are defined in the feature catalogue and must be encoded in the Feature Association field. Each relationship to another feature object is defined by:

1) The reference to the other feature object;
2) The association used for the relationship (Given by the code from the Feature catalogue);
3) The code of the role used within the association. Each association between the objects A and B has two roles, one for the relationship from A to B and one from the relationship from B to A.
   For example, the association ‘Aggregation’ has the roles: ‘Consists of’ and ‘Is part of’.

Note that only one direction of the relationship has to be encoded explicitly, the other direction is always implicit. For example an aggregation object has encoded the relationships to its parts but there is no explicit encoding for the relationships from the parts to the aggregation object. For each association a separate field has to be used. The association itself can have attributes. The attributes are encoded in the field by the same mechanism as described for the ATTR field. The same subfields are used at the end of the association field

Theme objects are a special kind of aggregation objects. They do not define an object itself, but group other objects together. The reasons for the grouping are mostly thematic; other reasons are possible. Each feature object may belong to more than one theme. Themes are therefore not mutually exclusive. Since the kind of association from a theme object to its members (and vice versa) is not variable, the encoding of this type of association is different from the other feature associations. A separate field, the Theme Association field is used. The association is always encoded from the feature object that belongs to the theme to the theme object itself.

If parts of the geometry are not intended to be used for the depiction of a feature object these spatial objects can be specified in the MASK field. Note that spatial objects may not be used directly by the feature object. For example, if a feature object is defined by a surface only, a curve that forms a part of the surface boundary can be masked.

The MASK field consists of a reference to a record and an update instruction.

Feature types may reference other feature types. For this encoding it is important that a Feature Type record must be stored prior to any record that references this record.

10a-7.3.2 Feature Type record structure

```
Feature Type record
   |
   |--FRID (5): Feature Type Record Identifier field
      |
      |-<0..1>-FOID (3): Feature Object Identifier field
      |
      |-<0..*>-ATTR (*5): Attribute field
      |
      |-<0..*>-INAS (5\\*5): Information Association field
      |
      |-<0..*>-SPAS (*6): Spatial Association field
      |
      |-<0..*>-FASC (5\\*5): Feature Association field
      |
      |-<0..*>-THAS (*3): Theme Association field
      |
      |-<0..*>-MASK (*4): Masked Spatial Type field
```

10a-7.3.2.1 Feature Type Record Identifier field structure

Field Tag: **FRID**	Field Name: Feature Type Record Identifier

Subfield name	Label	Format	Subfield content and specification
Record name	RCNM	b11	{100} – Feature type
Record identification number	RCID	b14	Range: 1 to 2^{32}-2
Numeric Feature Type Code	NFTC	b12	A valid feature type code as defined in the FTCS field of the Dataset General Information Record
Record version	RVER	b12	RVER contains the serial number of the record edition
Record update instruction	RUIN	b11	{1} – Insert {2} – Delete {3} – Modify

Data Descriptive Field

```
1100;&□□□Feature□Type□Record□Identifier▲RCNM!RCID!NFTC!RVER!RUIN▲(b11,b14,
2b12,b11)▼
```

10a-7.3.2.2 Feature Object Identifier field structure

Field Tag: **FOID**	Field Name: Feature Object Identifier

Subfield name	Label	Format	Subfield content and specification
Producing agency	AGEN	b12	Agency code
Feature identification number	FIDN	b14	Range: 1 to 2^{32}-2
Feature identification subdivision	FIDS	b12	Range: 1 to 2^{16}-2

Data Descriptive Field

```
1100;&□□□Feature□Object□Identifier▲AGEN!FIDN!FIDS▲(b12,b14,b12)▼
```

10a-7.3.2.3 Spatial Association field structure

Field Tag: **SPAS**	Field Name: Spatial Association

Subfield name	Label	Format	Subfield content and specification
Referenced Record name	*RRNM	b11	Record name of the referenced record
Referenced Record identifier	RRID	b14	Record identifier of the referenced record
Orientation	ORNT	b11	{1} – Forward {2} – Reverse {255} – NULL (Not Applicable)
Scale Minimum	SMIN	b14	The denominator of the smallest scale for which the referenced geometry can be used for the instance of the feature type (for example, for depiction) {2^{32}-1} NULL (Not Applicable) (See Note)
Scale Maximum	SMAX	b14	The denominator of the largest scale for which the referenced geometry can be used for the instance of the feature type (for example, for depiction) {2^{32}-1} NULL (Not Applicable) (See Note)
Spatial Association Update Instruction	SAUI	b11	{1} – Insert {2} – Delete

Note: For a correct handling of older data, robust parsers should consider both 0 and 2^{32}-1 as 'Not Applicable' for the SMIN and the SMAX sub-field*.*

Data Descriptive Field

```
2100;&□□□Spatial□Association▲*RRNM!RRID!ORNT!SMIN!SMAX!SAUI▲(b11,b14,b11,2
b14,b11)▼
```

10a-7.3.2.4 Feature Association field

Field Tag: **FASC**	Field Name: Feature Association

Subfield name	Label	Format	Subfield content and specification
Referenced Record name	RRNM	b11	Record name of the referenced record
Referenced Record identifier	RRID	b14	Record identifier of the referenced record
Numeric Feature Association Code	NFAC	b12	A valid code for the feature association as defined in the FACS field of the Dataset General Information Record
Numeric Association Role Code	NARC	b12	A valid code for the role as defined in the ARCS field of the Dataset General Information Record
Feature Association Update Instruction	FAUI	b11	{1} – Insert {2} – Delete {3} – Modify
Numeric Attribute Code	*NATC	b12	A valid attribute code as defined in the ATCS field of the Dataset General Information Record
Attribute index	ATIX	b12	Index (position) of the attribute in the sequence of attributes with the same code and the same parent (starting with 1)
Parent index	PAIX	b12	Index (position) of the parent complex attribute within this FASC field (starting with 1). If the attribute has no parent (top level attribute) the value is 0
Attribute Instruction	ATIN	b11	{1} – Insert {2} – Delete {3} – Modify
Attribute value	ATVL	A()	A string containing a valid value for the domain of the attribute specified by the subfields above

Data Descriptive Field

```
3600;&%/GFeature□Association▲RRNM!RRID!NFAC!NARC!FAUI\\*NATC!ATIX!PAIX!ATI
N!ATVL ▲(b11,b14,2b12,b11,{3b12,b11,A})▼
```

10a-7.3.2.5 Theme Association field

Field Tag: **THAS**	Field Name: Theme Association

Subfield name	Label	Format	Subfield content and specification
Referenced Record name	*RRNM	b11	Record name of the referenced record
Referenced Record identifier	RRID	b14	Record identifier of the referenced record
Theme Association Update Instruction	TAUI	b11	{1} - Insert {2} - Delete

Data Descriptive Field

```
2100;&□□□Theme□Association▲*RRNM!RRID!TAUI▲(b11,b14,b11)▼
```

10a-7.3.2.6 Masked Spatial Type field structure

Field Tag: **MASK**	Field Name: Masked Spatial Type

Subfield name	Label	Format	Subfield content and specification
Referenced Record name	*RRNM	b11	Record name of the referenced record
Referenced Record identifier	RRID	b14	Record identifier of the referenced record
Mask Indicator	MIND	b11	{1} – Truncated by the dataset limit {2} – Suppress portrayal
Mask Update Instruction	MUIN	b11	{1} – Insert {2} – Delete

Data Descriptive Field

```
2100;&□□□Masked□Spatial□Record▲*RRNM!RRID!MIND!MUIN▲(b11,b14,2b11)▼
```

The *Referenced Record identifier* field corresponds to the *spatialRef* attribute of S100_GF_MaskReference (Part 3, Table 3-14). The *Mask Indicator* field corresponds to the *maskIndicator* attribute of S100_GF_MaskReference.

Page intentionally left blank

S-100 – Part 10b

GML Data Format

Page intentionally left blank

Contents

Page intentionally left blank

10b-1 Scope

This Part specifies a profile of GML meant to be used as a basis for the development of GML application schemas for S-100 data products. The GML application schema for each S-100 data product defines a file format for the machine-to-machine exchange of information structured in conformance with the application schema for the data product and matching the content of the Feature Catalogue, as defined in the appropriate Product Specification.

The scope of this Part includes:

1) Feature and information type data conforming to the S-100 General Feature Model defined in Part 3, encoded using GML (ISO 19136), and structured as datasets (identifiable collections of data).
2) Construction of GML Schemas from Feature Catalogues.
3) Guidance for use of the Schemas in Application Schemas for data products.

The following are outside scope:

1) A format for updates to datasets.
2) Interchange by means other than datasets encapsulated as files, such as Web Feature Service (WFS), other Web services, email, etc.
3) Information not encapsulated using GML, such as Feature Catalogues, Exchange Set metadata, Portrayal Catalogues, and support files in other XML formats.
4) Tools for developing GML Application Schemas for data products.
5) Design and programming of software for processing GML data.
6) Gridded and coverage data.

10b-2 Conformance

The profile described in this Part conforms to the requirements for GML profiles described in ISO 19136.

10b-3 References

ISO 19106:2003, *Geographic information – Profiles*

ISO 19107:2003, *Geographic information – Spatial schema*

ISO 19111:2007, *Spatial referencing by coordinates (coordinate reference systems)*

ISO 19118:2005, *Geographic information – Encoding*

ISO 19123:2005, *Schema for coverage geometry and functions*

ISO 19136:2007, *Geographic information – Geography Markup Language*

ISO 19136-2:2015, *Geographic information – Geography Markup Language*

ISO/TS 19139, *Geographic information – Metadata – XML schema implementation*

ISO/IEC 19757-3, *Information technology – Document Schema Definition Languages (DSDL) – Part 3: Rule-based validation – Schematron*

IETF RFC 2396, *Uniform Resource Identifiers (URI): Generic Syntax*

IETF RFC 3986, *Uniform Resource Identifiers (URI): Generic Syntax*

W3C XLink, *XML Linking Language (XLink) Version 1.0, W3C Recommendation*

W3C XML Namespaces, *Namespaces in XML, W3C Recommendation*

W3C XML, *Extensible Markup Language (XML) 1.0, W3C Recommendation*

W3C XML Schema Part 1, *Structures, W3C Recommendation*

W3C XML Schema Part 2, *Datatypes, W3C Recommendation*

LEIRI, *Legacy Extended IRIs for XML Resource Identification, W3C Working Group Note 3*. URL: http://www.w3.org/TR/leiri

10b-3.1 Non-normative references

The following references are listed only for informative purposes or to clarify parts of this document. Drafts are subject to change and are not international standards.

ISO/DIS 19107, *Geographic information – Spatial schema* (Draft – June 2018)

10b-4 Introduction

The S-100 GML profile defines the core GML components that shall be used in GML encodings for S-100 data products. This profile defines a restricted subset of XML and GML types that excludes GML features not required by S-100 GML datasets. The profile of GML is contained in multiple Schema files and reduces the complexity of the full GML encoding to a more manageable level. Part of the Schema defines common elements and types needed for all S-100 datasets encoding feature-based information.

10b-5 General concepts

A GML application schema is an XML schema that conforms to the rules for application schemas given in the GML specification (ISO 19136).

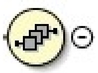
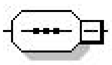
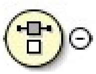
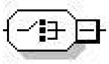
A GML document is an XML document with a root element conforming to the rules for GML data specified in the GML specification (ISO 19136). Specifically, in the context of S-100 this means the root element must be a GML AbstractFeature or Dictionary element, or in a substitution group of any of these elements.

The terms "GML application schema" and "application schema" as used in this Part mean respectively an *XML schema* and a *conceptual schema*. The former may be an XSD file conforming to the XML schema rules, the latter a UML diagram, conforming to the S-100 Application Schema rules.

These terms and definitions conform to ISO 19101 and ISO 19136. Complete definitions are given in ISO 19101 and ISO 19136, and are reproduced in Annex A.

An S-100 based Feature Catalogue presents the abstraction of reality represented in one or more sets of geographic data as a defined classification of phenomena. It presents a machine-readable description of a Product Specification Conceptual Schema according to S-100 Part 5.

10b-6 Notation and diagram conventions

Diagram element	Meaning
 or 	XML Schema <sequence>
 or 	XML Schema <choice>
0..∞	XML schema multiplicity constraints (here, "0" and "unbounded")

10b-7 Components and relationships to standards

The GML data encoding for S-100 consists of the following components, realized as separate XML schemas:

1) An XML schema that defines a GML profile (“**Profile”**). This is a restricted subset of types and elements (“XML constructs”) defined in the GML 3.2.1 schemas. XML constructs not needed for S-100 data products are excluded.
2) An XML schema defining additional XML constructs (“**S100base”**). This schema uses the GML profile schema. The constructs defined in this schema are expected to be needed in order for a product specification conforming to the S-100 standard to define a format for datasets.

Figure 10b-1 below illustrates the relationships between the various elements of the S-100 framework, an S-100 product specification and the GML profile. The GML encoding standard (ISO 19136:2007) provides an implementation schema (XML Schema) for the ISO 19100 conceptual schemas. The S-100 GML profile is a subset of the constructs defined by the GML implementation schema. S-100 common elements are defined in a common elements XML schema conforming to the profile. GML formats for specific data products use the constructs in the common elements to define XML types and elements corresponding strictly to feature and information types defined by the relevant Product Specification in the Feature Catalogue. A dataset is an XML file conforming to the GML Application Schema.

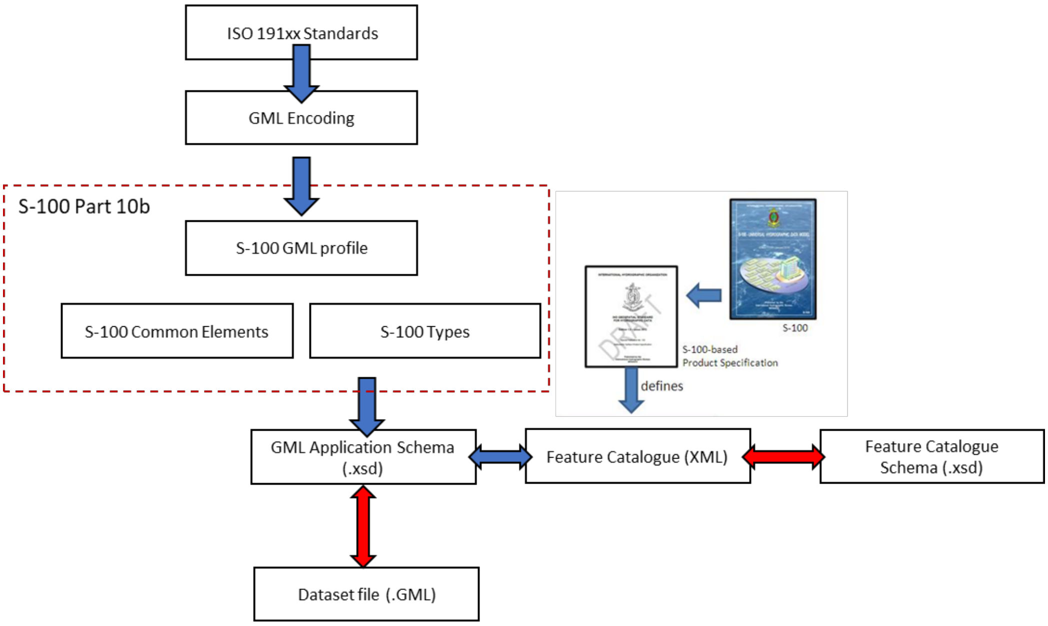


Figure 10b-1 – Derivation of profile and its use by a data product

10b-7.1 Use of profile

The typical use of the profile is to define a format for GML files encapsulating datasets packaged as files.

Formats for modes of exchange other than datasets are not required to use these Schemas.

The S-100 GML Profile has been defined to support Schema validation. The S-100 GML Profile must be declared within the Application Schema. This allows a validation engine to select this Schema instead of the GML 3.2.1 Schema to validate the data against.

A GML Application Schema mirrors the feature/attribute names, multiplicities, geometric primitives, types and relationships which are defined in the Feature Catalogue. While Schema validation is possible it should also be possible to unambiguously parse a GML dataset using the Feature Catalogue without recourse to the Application Schema.

10b-8 Profile for feature data

10b-8.1 Feature and information types

The profile supports the ability to encode classes defined as identifiable objects as derived from either the abstract GML type or abstract feature type:

- AbstractGML (shall be used to derive S-100 Information Classes).
- AbstractFeature (shall be used to derive S-100 Feature Classes).

The S-100 GML Profile prohibits the use of the gml:StandardObjectProperties group.

10b-8.2 Data types

10b-8.2.1 Primitive types

The S-100 GML Profile supports the types defined in S-100 Feature Catalogues (as defined in S-100 Part 2a, clause 2a-4.2.9) with the following equivalents. Types are supported using XML Schema (“xs:”) built-in data types where possible except as noted.

Table 10b-1 – S-100 primitive types

Feature Catalogue Types	XML / GML Profile Equivalent
boolean	xs:boolean
integer	xs:integer
real	xs:decimal
text	xs:string
date	xs:date
time	xs:time
dateTime	xs:dateTime
S100_TruncatedDate	S100_TruncatedDate
URI, URN, URL	xs:anyURI

All S-100 types referred to in Table 10a-1 are defined within the S-100 GML Profile. The profile also provides common restrictions of types for non-negative Decimals and Decimals in the range 0.0-360.0 (for ° representations).

10b-8.2.2 Extended Value types

The S-100 GML Profile supports the value types defined in Part 1, clause 1-4.5.3.5:

- Measure
- Length
- Angle

NOTE: S100_UnitOfMeasure type shall be realised by the uom property, the value of which should reference to a value defined in a codelist Register which provides the name, definition and symbol.

10b-8.2.3 Attribute types

S-100 simple attributes shall take one of the types defined in Table 10b-1 above. S-100 Complex attributes have no named type in the GFM.

The naming convention for Schema types of simple attributes and complex attributes shall have a defined type name “`<Name>Type`” where `<Name>` is the ”code” field defined in the associated Feature Catalogue. The “code” field is expected to be compliant with the xs:NCName criteria.

10b-8.2.4 Enumerations

For S-100 enumeration or S-100 codelist attributes, datasets must use the code and label of the listed value as encoded in the Feature Catalogue.

Enumerations shall be constructed using a combination of restricted simple types for the code and label.

The code simple type shall be a restriction of xs:integer with the list of allowable enumeration codes from the Feature Catalogue. The type name shall be constructed as `<Name>Code` using the attribute code.

```
<xs:simpleType name="cardinalDirectionCode">
  <xs:restriction base="xs:integer">
    <xs:enumeration value="1">
      <xs:annotation>
        <xs:documentation>N</xs:documentation>
      </xs:annotation>
    </xs:enumeration>
    <xs:enumeration value="2">
      <xs:annotation>
        <xs:documentation>NNE</xs:documentation>
      </xs:annotation>
    </xs:enumeration>
…
```

The label simple type shall be a restriction of xs:string using the allowable enumeration labels from the Feature Catalogue. The type name shall be constructed as `<Name>Label` using the attribute code.

```
<xs:simpleType name="cardinalDirectionLabel">
  <xs:restriction base="xs:string">
    <xs:enumeration value="N">
      <xs:annotation>
        <xs:documentation>1: north</xs:documentation>
      </xs:annotation>
    </xs:enumeration>
    <xs:enumeration value="NNE">
      <xs:annotation>
        <xs:documentation>2: northnortheast</xs:documentation>
      </xs:annotation>
    </xs:enumeration>
…
```

The enumeration attribute type is defined as a complex type extending from the label type and including the code type as an xml attribute.

```
<xs:complexType name="cardinalDirectionType">
  <xs:annotation>
    <xs:documentation>Cardinal direction: Principal and intermediate compass points.</xs:documentation>
  </xs:annotation>
  <xs:simpleContent>
    <xs:extension base="cardinalDirectionLabel">
      <xs:attribute name="code" type="cardinalDirectionCode"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

10b-8.3 Feature collections

A feature collection is a collection of feature instances. Within GML 3.2.1, the generic gml:FeatureCollection element has been deprecated. A feature collection is any feature class with a property element in its content model which is derived by extension from *gml:AbstractFeatureMemberType*.

The S-100 GML Profile supports the GML 3.2.1 approach to model a Feature Collection class within an S-100 GML Application Schema.

For exchange of file-based GML data using S-100 the GML Profile includes:

- A FeatureCollection, named “Dataset”; and
- Generic dataset metadata types with mandatory dataset metadata elements

Other exchange mechanisms, for example via API mechanisms may wrap GML data encoded in conformance with this part with different feature collection mechanisms.

10b-8.4 Associations

The profile allows associations to be encoded inline or by reference. The dataset metadata field associationEncoding shall be defined as either “reference” or “inline” to define which method is used throughout conforming datasets. In addition to the dataset metadata constraint such associations shall only be used to express composition relationships defined by the corresponding Feature Catalogue.

For bi-directional associations, the profile supports the optional encoding of the name of reverse property in the *appInfo* annotation element in the Application Schema XSD.

10b-8.4.1 Association classes

The profile allows the GML 3.3 convention for encoding of association classes using the GML 3.3 association class conversion rule, which converts association classes to an equivalent intermediate class. The figures below illustrate the conversion rule.

Where associations contain attributes in a Product Specification Feature Catalogue, this structure shall be used to realise those attributes via an intermediate Information Type. The name of the Information Type shall be defined as the name (S100_FC_Item code) of the relationship (either feature or information association) concatenated with ”Type” and is not required to be defined within the Feature Catalogue.

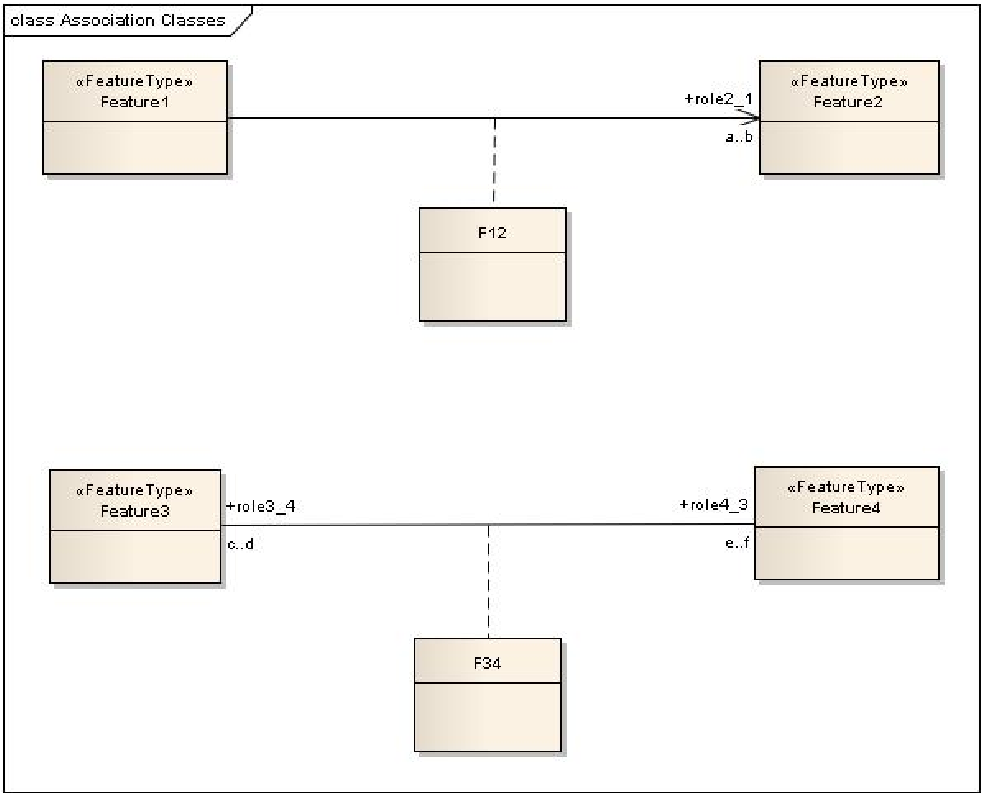


Figure 10b-2 – Model with association classes (from OGC 10-129r1 / ISO 19136-2:2015)

An example (taken from S-127) is shown in Figure 10b-3 below. Here an association class “PermissionType”, corresponding to a relationship with code “Permission”, is used to express the association attribute “categoryOfRelationship”.

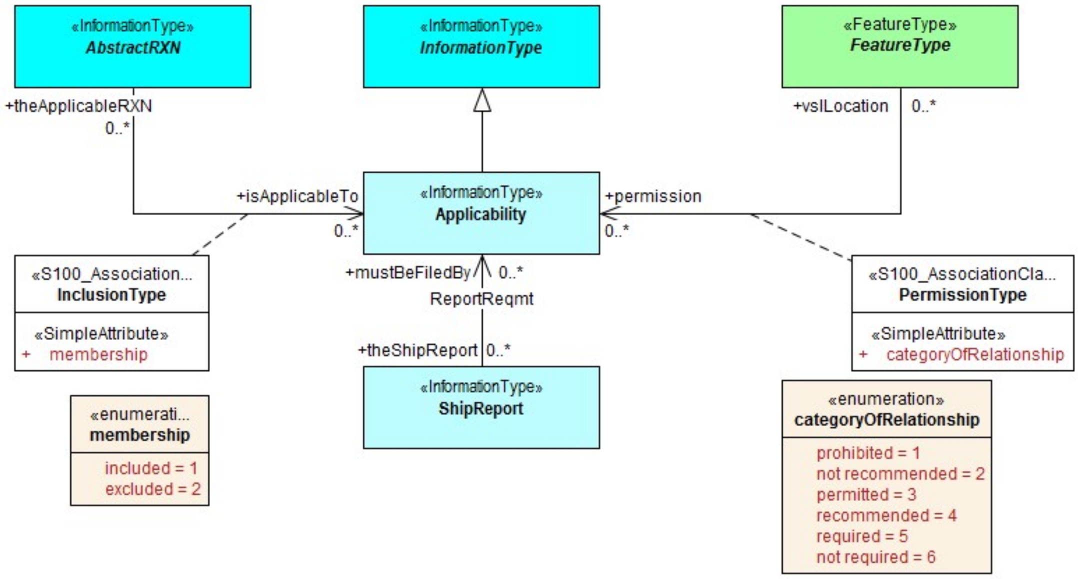


Figure 10b-3 – Example use to represent attributes of associations (S-127)

To avoid needless duplication of relationships between source/destination features and the intermediate information types, the intermediate information type shall be included in line with the referencing feature; for example:

```
<S-127:VesselTrafficServiceArea gml:id="a1">
  <categoryOfCargo>ballast</categoryOfCargo>
  <permission xlink:href="#R1">
    <S-127:PermissionType gml:id="res1">
      <categoryOfRelationship>required</categoryOfRelationship>
    </S-127:PermissionType>
  </permission>
</S-127:VesselTrafficServiceArea>
<S-127:Applicability gml:id="R1">
  <categoryOfCargo code=”7”>dangerous or hazardous</categoryOfCargo>
</S-127:Applicability>
```

In the Application Schema this shall be done by extension of gmlReferenceType.

```
<xs:complexType name="PermissionType">
  <xs:sequence>
   <xs:element name="categoryOfRelationship" type="categoryOfRelationshipType" minOccurs="1"
   maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>
        <xs:complexType name="permission">
                <xs:complexContent>
                        <xs:extension base="gml:ReferenceType">
                                <xs:sequence>
           <xs:element name="PermissionType" type="PermissionType" minOccurs="1" maxOccurs="1"/>
                                </xs:sequence>
                        </xs:extension>
                </xs:complexContent>
        </xs:complexType>
```

The examples above are informative, not mandatory.

10b-9 S-100 GML data formats

Data formats shall use the camel case codes of features, information types, and attributes and sub-attributes as specified in Feature Catalogues as the ‘local name’ in element tags for GML features and attributes.

EXAMPLE: Given a Feature Catalogue that defines a feature named “Marine Protected Area” with code “MarineProtectedArea”, the corresponding feature in the dataset must use “MarineProtectedArea” as the local name – for example, `<S122:MarineProtectedArea ...` or `<MarineProtectedArea>` with the Schema type “MarineProtectedAreaType”

Ordering of elements in GML Schemas corresponding to simple and complex attributes shall be identical with ordering in the Feature Catalogue. Attributes and sub-attributes inherited from abstract types in the Feature Catalogue shall appear before those belonging to the specialised type.

Spatial objects that are encoded independently of features (that is, not embedded inline with a feature) shall be encoded with tags whose local name components are the spatial object elements in the S-100 GML profile (for example, S100:Point).

The GML Profile defines abstract types for feature and information types to be used for definition of concrete types contained in the Feature Catalogue.

Feature and information associations must encode at least one of the *role* or *arcrole* attributes of the reference.

The following tags are reserved and may not be used in GML data formats as local names of elements:

- geometry;
- Dataset.

GML data formats for S-100 datasets must follow the GML rules as described in the GML specification (ISO 19136/OGC 07-036), as modified by the S-100 GML profile and this Part.

10b-10 Processing of GML datasets

Implementations, including applications and production tools, may use any suitable method for processing GML datasets. While GML datasets must conform to the GML Application Schemas defined in Product Specifications, processors are not required to use the GML Application Schemas for processing datasets. However, the combination of the GML specification, this Part, and the S-100 GML profile result in the following commonalities:

1) Each dataset has a single root element (“Dataset”). GML datasets are XML documents and this is an XML requirement.
2) Given the path /Dataset /X1/X2 then X1 is a feature and X2 is an attribute or association role. Similarly, given /Dataset /X1/X2, X1 is an information type and X2 one of its attributes or associations.
3) If X2 has XML attributes xlink:href and xlink:role and/or xlink:arcrole it is an association role.
4) If X2 has element content this represents a complex or spatial attribute.
5) A spatial attribute or object will have one of the allowed spatial properties as its content.
6) If X2 is empty and nilled, or has text or numeric content, it is a simple attribute.
7) Applications must allow for the presence or absence of namespaces; for example, X1 might be of the form S122:FeatureA, etc. Namespaces in XML precede a ':' so it is possible for applications to distinguish the namespace part of the tag from the ‘local name’ part.

10b-10.1 Spatial types

10b-10.1.1 Geometric primitives

The S-100 GML Profile supports the S-100 implementations of GML 3.2.1 basic geometries (Part 7, clause 7-4.1.1).

For the simplest geometry requirements where no spatial attribution or associations are required, native GML types, imported by the S-100 GML Profile Schemas, and representing S-100 point, curve and surface geometry can be used. These implement equivalents to point, curve and surface geometry defined as geometric primitives in Feature Catalogues.

- **GM_Point**
- **GM_Curve**
- **GM_Surface**

More complex Schemas requiring referencing and spatial attributes (and more complex curve interpolations) can make use of the more complex constructions defined specifically for the S-100 GML Profile;.for example:

- **S100_Point**
- **S100_Curve**
  - AbstractCurve
  - OrientableCurve
  - LineStringSegment
  - LineString
  - S100_ArcByCenterPoint
  - S100_CircleByCenterPoint
- **S100_Surface**

The S-100 GML Profile constrains the GM_CurveInterpolation type values and constrains the curve encoding to a subset of GML curve geometries.

Note: S100_ArcByCenterPoint and S100_CircleByCenterPoint are not the same as the GML primitives ArcByCenterPoint and CircleByCenterPoint.

10b-10.1.2 Curve Interpolation

The list of allowable values consists of a subset of the values allowed by ISO 19136 plus extensions for spline and interpolated curve segments (ISO/DIS 19107 draft – June 2018, clarifies that the list of interpolations in the standard is not exhaustive):

1) **Linear (linear) in a non-geographic CRS** – the interpolation is defined by a series of DirectPositions on a straight line between each consecutive pair of controlPoints.
2) **Linear interpolation (linear) in a geographic CRS (interpreted as Loxodromic)** – the interpolation method shall return DirectPositions on a loxodromic curve between each consecutive pair of controlPoints. A loxodrome is a line crossing all meridians at the same angle, that is, a path of constant bearing.
3) **Geodesic (geodesic)** – the interpolation mechanism shall return DirectPositions on a geodesic curve between each consecutive pair of controlPoints. A geodesic curve is a curve of shortest length. The geodesic shall be determined in the coordinate reference system of the *GM_Curve* in which the *GM_CurveSegment* is used.
4) **Circular arc by 3 points (circularArc3Points**) – the interpolation is defined by a series of 3 DirectPositions on a circular arc passing from the start point through the middle point to the end point for each set of three consecutive control points. The middle point is located halfway between the start and end point.
5) **Elliptical arc (elliptical**) – for each set of four consecutive controlPoints, the interpolation mechanism shall return DirectPositions on an elliptical arc passing from the first controlPoint through the middle controlPoints in order to the fourth controlPoint. Note: if the four controlPoints are co-linear, the arc becomes a straight line. If the four controlPoints are on the same circle, the arc becomes a circular one.
6) **Conic arc (conic)** – the same as elliptical arc but using five consecutive points to determina a conic section.
7) **Circular arc with centre and radius (circularArcCenterPointWithRadius)** – the interpolation is defined by an arc of a circle of the specified radius centred at the position given by the single control point. The arc starts,at the start angle parameter and extends for the angle given by the angular distance parameter. This interpolation type shall be used only with S100_ArcByCenterPoint and S100_CircleByCenterPoint geometry. The precise semantics of the parameters are defined in Part 7 clause 7-4.2.20 (S100_ArcByCenterPoint).
8) **Polynomial (polynomialSpline)** – the control points are ordered as in a line-string, but they are spanned by a polynomial function. Normally, the degree of continuity is determined by the degree of the polynomials chosen.

9) **Bézier Spline (bezierSpline)** – the data are ordered as in a line string, but they are spanned by a polynomial or spline function defined using the Bézier basis. Normally, the degree of continuity is determined by the degree of the polynomials chosen.

10) **B-spline (bSpline)** – the control points are ordered as in a line string, but they are spanned by a polynomial or rational (quotient of polynomials) spline function defined using the B-spline basis functions (which are piecewise polynomials). The use of a rational function is determined by the Boolean flag "isRational". If isRational is TRUE then all the DirectPositions associated with the control points are in homogeneous form. Normally, the degree of continuity is determined by the degree of the polynomials chosen.

11) **Blended parabolic (blendedParabolic)** – the control points are ordered as in a line-string, but are spanned by a function that blends segments of parabolic curves defined by triplet sequences of successive data points. Each triplet includes the final two points of its predecessor. Further details of the semantics are provided in Part 7 clause 7-4.2.2.2.

10b-10.1.3 Geometric complex, geometric composites, and geometric aggregates

10b-1.1.1.1 Geometric complex and geometric composites

The S-100 GML Profile supports the following composite geometries (Part 7 clause 7-4.1.1):

- CompositeCurve.

10b-1.1.1.2 Geometric aggregates

The S-100 GML Profile supports the aggregate geometry types (Part 7 clause 7-4.1.1):

- MultiPoint

10b-10.1.4 Inline and by-reference encoding

The S-100 GML Profile supports the ability to encode a geometry either inline or by reference where two features share the same instance of a GM_Object (see Part 3 clause 3-6.5.4.5). Where both are specified the GML convention is followed and the inline reference takes priority.

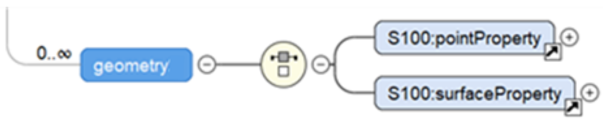
10b-10.1.5 Envelope

The S-100 GML Profile supports the ability to encode an appropriate geometry via bounding box or envelope. The Profile does not constrain the use of the GML implementation of GM_Envelope.

10b-10.1.6 Schema elements for spatial attributes

GML Application Schemas shall name the S100 spatial attribute type as geometry elements using the reserved element name "geometry". This is for greater interoperability with broader GML tools. Where multiple geometric primitive types are allowed by the Feature Catalogue defining the GML Application Schema these shall be implemented as XML Schema choice elements and not as aggregated types. Geometry elements shall have multiplicity 0..* in the GML Application Schema.

For example:



10b-10.1.7 Masking, truncation and scale ranges

The S-100 geometry type has an optional sub-element mask containing those elements which are masked.

Beginning with Edition 5.0.0, the S-100 GML format defines a generic complex type S100_SpatialAttributeType for spatial attributes with *scaleMinimum* and *scaleMaximum* attributes and a *maskReference* tag. These correspond to the attributes of the S100_SpatialAttribute metaclass in the S-100 General Feature Model (Part 3, Figures 3-1 and 3-2 and clause 3-5.3.5). *scaleMinimum* and *scaleMaximum* are implemented as integer attributes. The *maskReference* attribute is implemented using the GML Reference Type with the following constraints:

- The value of the xlink:href attribute must be the gml:id of the masked/truncated object.
- The value of the xlink:role attribute must be either 'truncated' or 'suppressed'.

- The meaning of other attributes is undefined in S-100 and so they are not used within the GML profile.

The structure of the S100_SpatialAttribute Complex type is depicted in Figures 10b-4 and 10b-5 below.

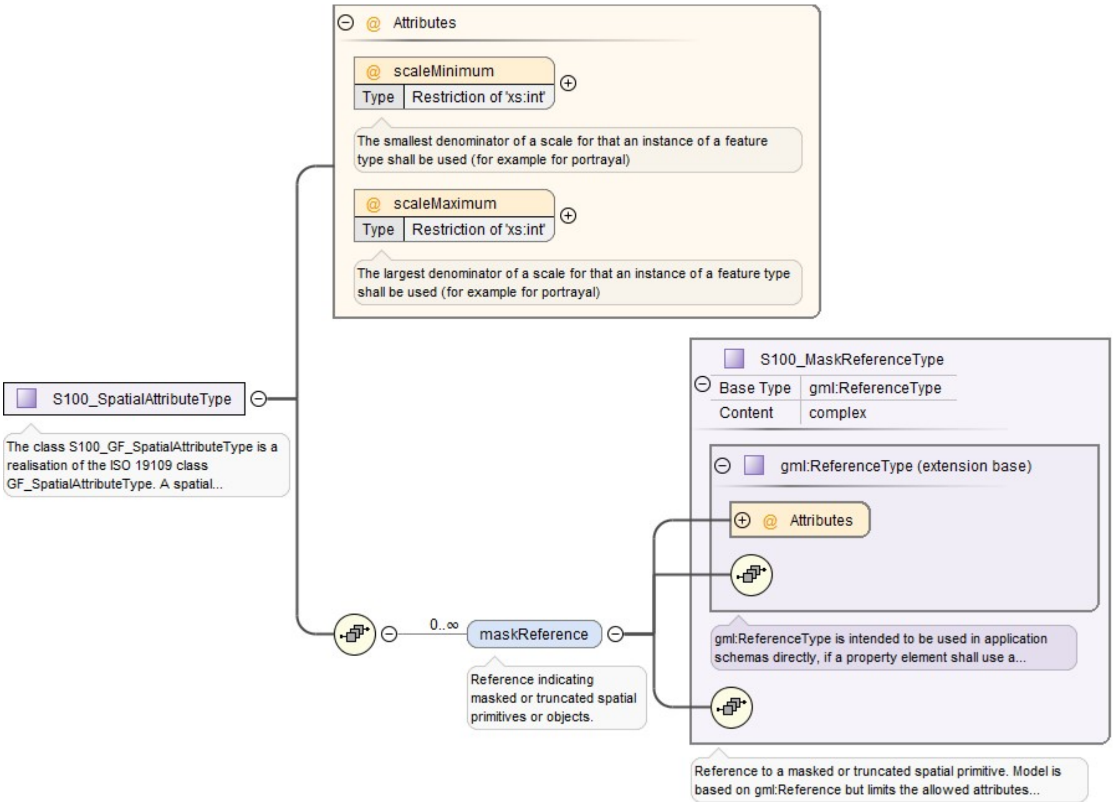


Figure 10b-4 – Structure of generic spatial attribute type in the S-100 GML format

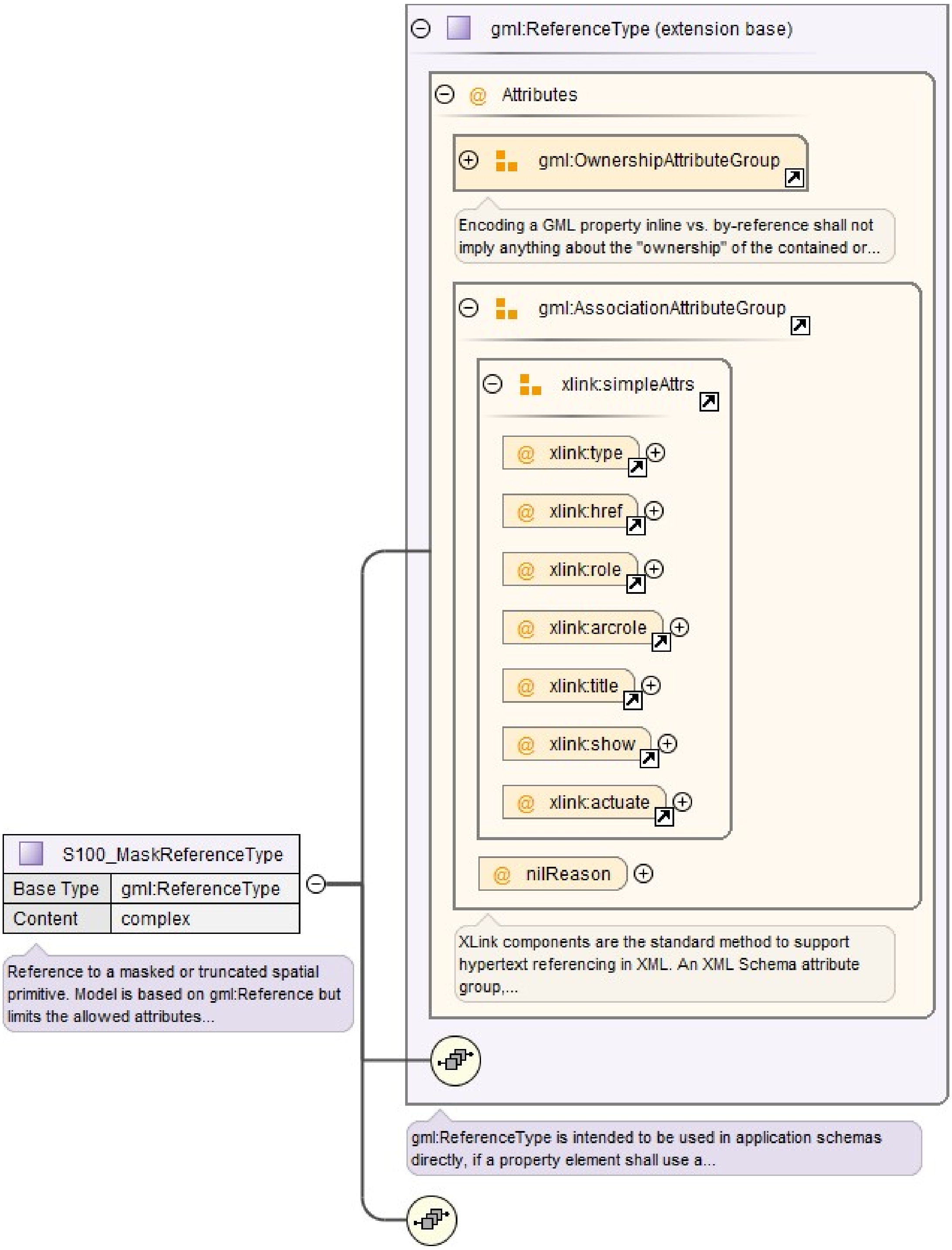


Figure 10b-5 – Structure of mask reference type

NOTE: This Part specifies only the *href* and *role* attributes. The other *AssociationAttributeGroup* members, *OwnershipAttributeGroup* members, and *nilReason* are not used.

An example of the use of masking is depicted in Figure 10b-6 below. The surface boundary is defined by reference to two curves (sequential curves comprising the exterior ring), whose gml:id's are JS.C.123 and JS.C.567. These curves are defined elsewhere in the file. The *maskReference* tag in the example indicates that the curve JS.C.567 is suppressed.

NOTE 1: The S-100 GML format does not require that the object geometry (the Surface object in Figure X.X) be encoded inline as depicted in Figure X.X. It can be encoded elsewhere in the dataset as a separate spatial data object, like the curves.

NOTE 2: The ‘#’ character preceding the identifier is an XML convention indicating that the part which follows is the identifier of an XML element inside an XML file (since no filename is specified, the convention is that the referenced element is in the same file). Note that the reference mechanism also allows references to objects in external files by prefixing the object identifier with the file name or URL of the external file.

```
<S127:MilitaryPracticeArea gml:id="JS.MIPARE.676">
    <featureName><name>Jussland Naval Firing Area</name></featureName>
    <textContent><information><text>Example of masked spatial type</text></inform
    <theServiceHours xlink:href="#JS.SRVHRS.685"/>
    <geometry>
        <S100:surfaceProperty scaleMinimum="1000000" scaleMaximum="10000">
            <S100:maskReference xlink:href="#JS.C.567" xlink:role="suppressed"/>
            <S100:Surface gml:id="JS.MIPARE.676.S.1"  srsDimension="2" srsName="u
                <gml:patches><gml:PolygonPatch><gml:exterior>
                    <gml:Ring>
                        <gml:curveMember xlink:href="#JS.C.123"/>
                        <gml:curveMember xlink:href="#JS.C.567"/>
                    </gml:Ring>
                </gml:exterior></gml:PolygonPatch></gml:patches>
            </S100:Surface>
        </S100:surfaceProperty>
    </geometry>
    <geometry>
```

Figure 10b-6 – Example of masking in the GML format

10b-10.2 Unsupported GML functionality

Support for GML 3.2.1 and GML 3.3 geometries not defined in ISO 19107 is not included. Specifically, this means CircleByCenterPoint and ArcByCenterPoint (as defined in GML 3.2.1) are not supported, nor are the compact geometry encodings defined in GML 3.3.

The temporal model and temporal primitives defined in ISO 19108, including temporal positions, instants, time periods, are not supported. S-100 data should code dates and times as thematic attributes.

- Dynamic features are not supported by the S-100 GML profile.
- Topology is not supported by the S-100 GML profile.
- Linear Referencing is not supported by the S-100 GML profile.
- Coverages are not supported by the S-100 GML profile.
- The ability to define coordinate reference systems is not supported. The products should be defined using a well-known, pre-defined coordinate reference system such as WGS84.
- Observations are out of scope for the S-100 GML Profile. (The observations schema within GML has been superseded by the OGC (10-025r1) XML encoding for ISO 19156: Observations and Measurements.)

10b-10.3 Compliance levels

In order for a client to be able to properly interpret a schema, it needs a capability to identify the compliance level of the application schema. An XML Schema annotation shall be used for this purpose. The following schema fragment shows how this annotation shall be declared in an application schema[1]:

[1] Line breaks and spaces have been added for clarity.

```
<annotation>
 <appinfo>
  <gmlProfileSchema xmlns="http://www.opengis.net/gml/3.2">
     http://www.iho.int/S-100/profiles/s100_GMLProfile.xsd
  </gmlProfileSchema>
  <s100:ComplianceLevel>1</s100:ComplianceLevel>
 </appinfo>
</annotation>
```

Table 10b-2 – Compliance declaration XML code

Compliance level	Description
1	S-100 feature types, information types, feature and information associations. Point, curve, and surface primitives
2	All features of Level 1, plus circle and arc by center point geometry, splines, and blended interpolations

To manually add the compliance declaration to the schemas after they have been generated involves 3 steps:

1. Add the S-100 GML Profile XML Namespace declaration:

   ```
   xmlns:s100_profile="http://www.iho.int/S-100/profile/s100_gmlProfile"
   ```

2. Add the S-100 GML Profile compliance declaration within the schema annotation. The compliance declaration is the XML code in Table 10b-1 above.

3. Add an Import statement for the S-100 GML Profile Levels schema. Add the following import statement for the S-100 GML Profile Levels schema into the list of imported schemas to the list of imported schemas:

   ```
   <import namespace="http://www.iho.int/S-100/profile/s100_gmlProfile"
          schemaLocation="../../S100/profile/S100_gmlProfileLevels.xsd"/>
   ```

10b-11 S-100 base schema for feature data

10b-11.1 Introduction

A second XML Schema is provided which defines a small set of derived types and elements in an "S100" namespace. The Schema defining these common elements and types is technically a "GML Application Schema" in the sense defined by ISO 19136. It defines GML constructs which are expected to be used by different Product Specifications to define detailed GML Application Schemas encoding formats for GML datasets. This Schema provides a common core structural paradigm for GML datasets across a variety of application domains. The intention is to reduce the complexity of application development, facilitate sharing of software modules, and information integration and mapping across different application domains, by minimising the proliferation of structural variations. The XML Schemas are designed to ensure a match between the Feature Catalogue structure and the produced GML Schema.

Elements and types are defined using only the restricted subset of GML defined in the S-100 GML Profile.

10b-11.1.1 Construction of the xsd header

The Application Schema header must define namespaces and prefixes for all imported elements of the S-100 GML Profile. An example top level Schema element is shown below:

```
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:S100="http://www.iho.int/s100gml/1.0"
xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns="http://www.iho.int/S123/gml/1.0"
```

xmlns:S100EXT="http://www.iho.int/s100gml/1.0+EXT"
targetNamespace="http://www.iho.int/**S123**/gml/1.0"
elementFormDefault="qualified"
version="1.0.0-20170831">

The namespaces defined are:

xmlns:gml=http://www.opengis.net/gml/3.2
xmlns:S100="http://www.iho.int/s100gml/1.0"
xmlns="http://www.iho.int/S123/gml/1.0"
targetNamespace=http://www.iho.int/S123/gml/1.0
elementFormDefault="qualified"

The Schema default namespace and the target namespace are the same and should be formed using a consistent pattern defined in S-100 Part 10b; perhaps based on the Product Specification and the Product Specification version. Using elementFormDefault = 'qualified' means every element in the dataset must belong to a namespace but it does not mean that a namespace prefix is needed; a default namespace can be used to set the namespace for any elements without a specific prefix.

NOTE: If the dataset is meant to contain more than one product then no default is used and every element requires a namespace or namespace prefix.

10b-11.2 Feature and information types

An XML complex type AbstractFeatureType is defined as the base types for all geographic features in the S-100 profile. Beginning with S-100 Edition 5.0.0 no generic associations are defined within the abstract feature types or information types.

All associations shall be implemented as named associations with names and types corresponding to those used in the source Feature Catalogue. The GML identifier gml:id may shall be used as a default feature identifier.

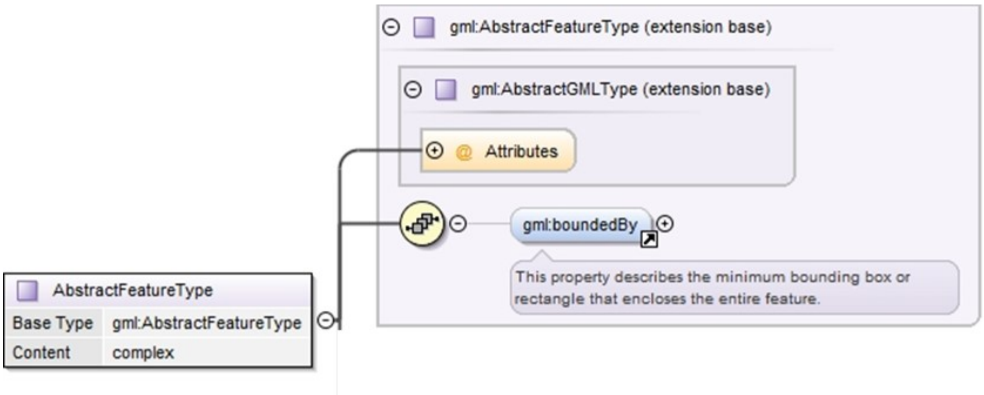


Figure 10b-7 – Base type definition for S-100 feature elements

The common type definition for information types is similar to that for the common feature type.

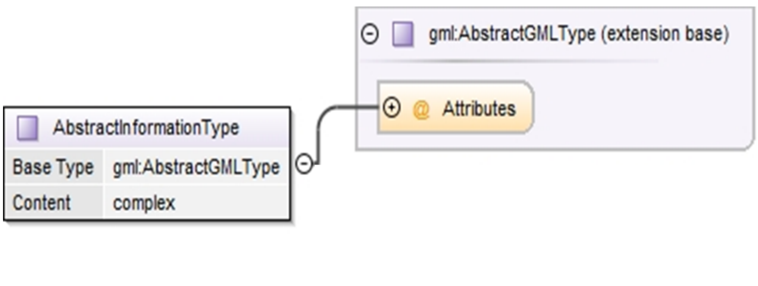


Figure 10b-8 – Base type definition for S-100 information type elements

10b-11.3 Spatial types

More complex spatial types are defined as extensions of the corresponding GML spatial types with an information association added, since in S-100 spatial objects can have information associations. The Figure below shows the design of the Point type in the S-100 schema. It includes a single gml:Point type and 0 or more associations to S-100 information types.

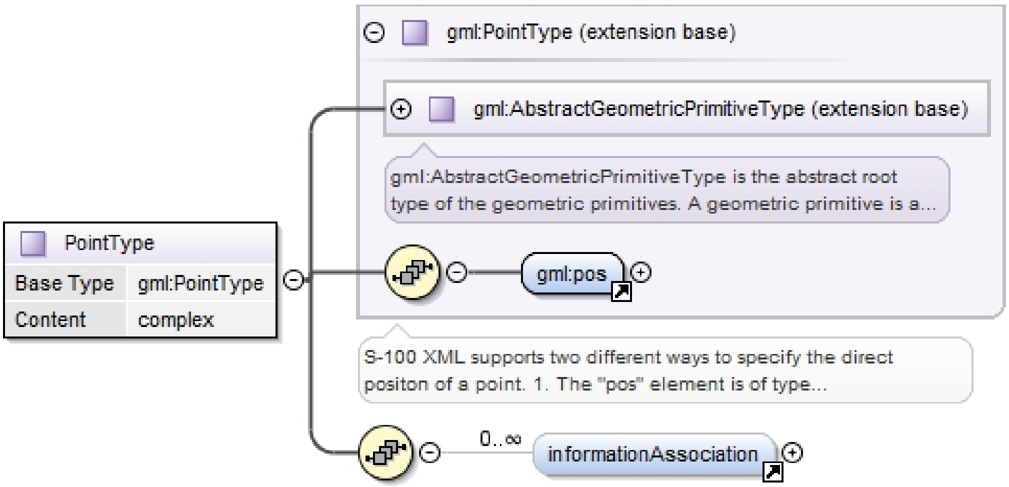


Figure 10b-9 – S-100 generic S-100 spatial type for point spatial object

The other spatial types have a similar structure.

10b-11.3.1 Inline and referenced geometry

The base schema also allows geometry to be defined either inline or by reference, conforming with the same ability in GML. Where both are included in a dataset, the inline geometry is normative.

10b-11.3.2 Spatial types defined in base schema

The base schema defines the point, curve, and surface spatial objects, as well as multipoint and composite curve objects. Curves may be simple, composite, or orientable curves. This is the same set defined in the S-100 GML profile (clause 10b-10.1). It also supports *gml:Polygon* (ISO 19136) which is a special surface that is defined by a single surface patch.

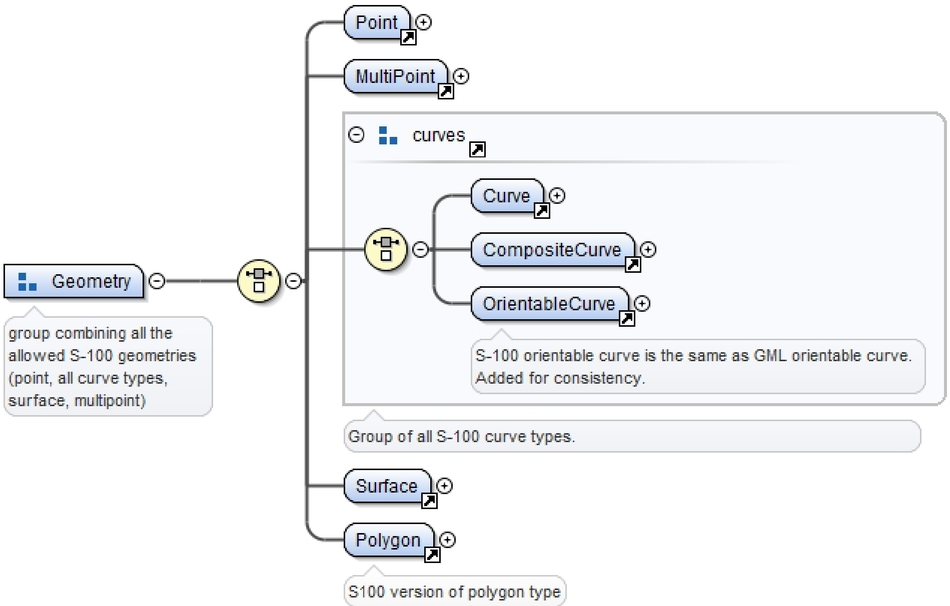


Figure 10b-10 – Geometry types defined in the base schema

10b-11.4 Associations

Feature and information association properties are defined as extensions of GML feature properties. The pointers to the object at the other end of the association are encoded in the Xlink attributes.

XLink components are the standard method to support hypertext referencing in XML.

GML provides an XML Schema attribute group, *gml:AssociationAttributeGroup*, to support the use of Xlinks as the method for indicating the value of a property by reference in a uniform manner in GML. This structure shall be used to encode all associations in the associated Feature Catalogue.

ISO 19136 specifies that the value of a GML property that carries an *xlink:href* attribute is the resource returned by traversing the link.

The data types of the attributes are listed in the Table below.

Table 10b-3 – Requirements for XLink attributes in associations

Xlink attribute	Data type	Remarks
href	URI	Reference to the object at the other end of the association, for example, the gml:id of an object in the current data set. May be a URI fragment.as described in the XLink specification
role	URI	Optional description of the nature of the target resource, given as a URI
arcrole	legacy extended IRI	Description of the role or purpose of the target resource in relation to the present resource, given as a URI (ISO 19136). May be constructed from the role name from the application schema. The XLink 1.1 specification requires: 1) The value must be a Legacy extended IRI 2) The identifier must not be relative
title	character string	Optional string describing the relationship. Product specifications may constrain its format and define its semantics
show		not used
actuate		not used
type		not used

Beginning with S-100 Edition 5.0.0, Application Schemas may only use named reference types for associations between features and information types, using the role name as a property element, with generic associations deprecated for use in GML Application Schemas.

10b-11.4.1 Role name as property element

The roles defined in the Application Schema shall be used as the property element of the feature or information type with XLink attributes providing the reference to the instance. In this case the role at the far end of the association should be used for the XML tag defining the property. The role name may be usable as-is for the property tag, or it may have to be mapped to a tag conforming to XML and GML conventions.

Example (non-normative): Given an Application Schema containing the relationship in the figure below, the *NavigationLine* feature can encode the association as a property element named *navTrack* as below. The format, construction rules, and semantics for the *arcrole* and *title* values would be defined in the Application Scheme.

```
<NavigationLine gml:id="US123098">
  <navTrack xlink:href="#US890321"
        xlink:arcrole="urn:iho:s101:1.0:52.2" title="RangeSystem"/>
  ...
</NavigationLine>
```

and elsewhere in the same file:

```
<RecommendedTrack gml:id="US890321">
  <navLine xlink:href="#US123098"
```

```
        xlink:arcrole=”urn:iho:s101:1.0:52.1” title=”RangeSystem”/>
    ...
</RecommendedTrack>
```

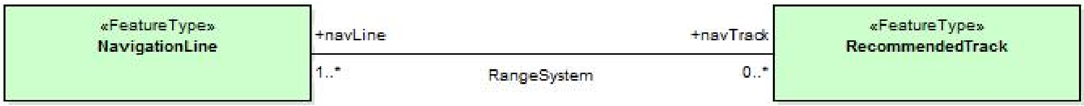


Figure 10b-11 – Association in Application Schema

10b-11.5 Updating

Datasets may have two purposes.

1. Base datasets containing all features, information types and associations within a specific coverage area, for a given dataset issue date.
2. Numbered update datasets, containing only updates to individual features within an earlier base dataset.

Updates are numbered, with a sequential update number, starting at 1, listed in the dataset metadata. A dataset’s purpose is listed in the dataset metadata element and in the catalogue metadata for the exchange set in which it is contained.

Updates are only defined at a feature or information type level. They can only add new features or replace earlier definitions of individual dataset features. Updated features shall retain the GML identifier issued in the base dataset. It is not possible to update associations except by replacing all features affected by a change to references. It is not possible to delete features through update.

10b-11.6 Dataset general information

10b-11.6.1 Dataset identification

Dataset identification information is defined by the complex type *DatasetIdentificationType*. The fields are shown in Table 10b-4 and Figure 10b-12 below.

Table 10b-4 – Dataset identification header elements

Field	XML Tag	Value	Mult.	Type	Description
Encoding specification	encodingSpecification	‘S-100 Part 10b’	1	CharacterString	Encoding specification that defines the encoding
Encoding specification edition	encodingSpecificationEdition	“1.0”	1	CharacterString	Edition of the encoding specification
Product identifier	productIdentifier		1	CharacterString	Unique identifier for the data product
Product edition	productEdition		1	CharacterString	Edition of the product specification
Application profile	applicationProfile		1	CharacterString	“1” – base datasets “2” – update datasets
Dataset file identifier	datasetFileIdentifier		1	CharacterString	The file name including the extension but excluding any path information
Dataset title	datasetTitle		1	CharacterString	The title of the dataset
Dataset reference date	datasetReferenceDate		1	date	The issue date of the dataset.

					Format: YYYY-MM-DD
Dataset language	datasetLanguage	“ENG”	1	ISO 639-1-2/T	The (primary) language used in this dataset 3 character codes
Dataset abstract	datasetAbstract		0..1	CharacterString	The abstract of the dataset
Dataset topic category	datasetTopicCategory	{14}{18}	1..*	MD_TopicCategoryCode (ISO 19115-1)	A set of topic category codes from the MD_TopicCategoryCode list in ISO 19115-1 (except “extraTerrestrial”)
Dataset Purpose	datasetPurpose	{“Base”,”Update”}	1	CharacterString	Whether dataset consists of updated features or all features
Update Number	updateNumber		1	Integer	The sequential update number of this dataset

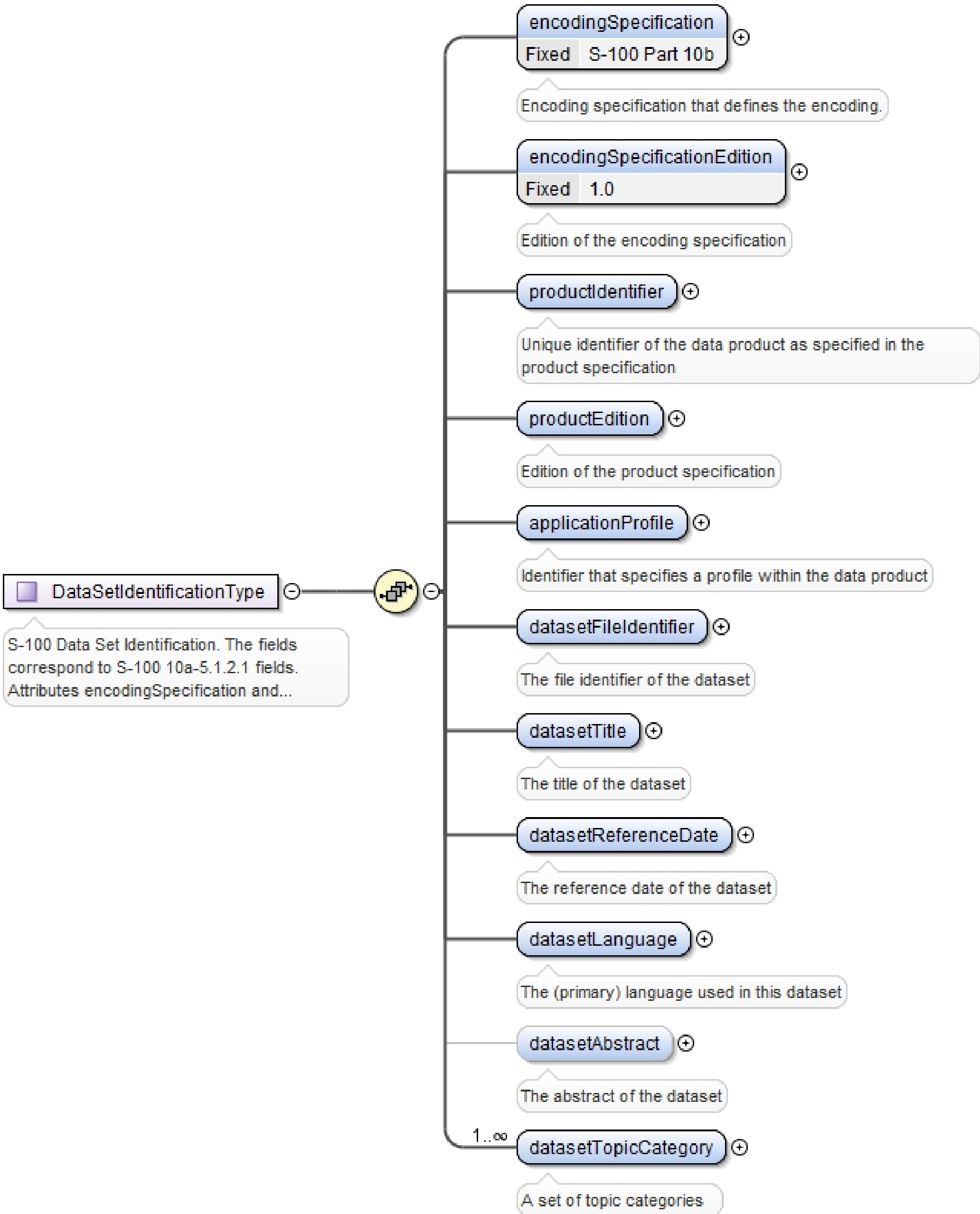


Figure 10b-12 – Dataset identification

10b-11.7 Coordinate Reference System

GML allows the coordinate reference system ("spatial reference system") used for geometry to be determined in different ways – by explicit specification, or by "inheriting" the SRS of outer elements. For S-100 datasets, this means the SRS can be specified in one of two ways:

- Using the srsName attribute of the gml:Envelope element in a feature collection implies that the same SRS is uses for all geometries contained in that collection.
- Using the srsName and srsDimension attributes for individual geometry elements.

Application data formats may use either method, but shall ensure that the SRS of every instance of geometry in a dataset can be determined by application software, using one method or another.

"Standard" geodetic coordinate reference systems shall be identified using the URI convention for SRS specified by OGC.

EXAMPLE `http://www.opengis.net/def/crs/EPSG/0/4326`

10b-11.8 Dataset structure definition

Application Schemas for data products shall define an XML type and element to serve as the root element of a GML dataset, consisting of a collection of XML elements for feature, information type, and spatial data objects defined elsewhere in the Application Schema. This shall be called “Dataset”.

10b-11.8.1 Dataset metadata

The dataset class may contain one or metadata properties to encode dataset level metadata (for example ISO 19115/19139) either inline or by reference.

10b-12 Constraints and validation

Some validation of data can be done using validating XML processors if the data product’s GML application schema created well defined types wherever possible, for example, enumerated types for the enumeration attributes, and maximum and minimum allowed values for real attributes. However, complete validation, especially of conditional attributes, is likely to require an additional means of data validation.

Constraints allow complex business rules to be defined that restrict the allowable values based on well-defined limits or relationships between properties (for example the end date must be equal to or greater than the start time).

Constraints can be defined in many different ways - human readable text only, object constraint language (OCL), Semantics of Business Vocabulary and Business Rules (SVBR) and these can be documented as part of the UML model or external to the model.

The S-100 GML Profile does not provide explicit support for expressing constraints or for rule-based validation. Current industry best practice, advocates the uses of Schematron to validate XML files based against business rules defined using OCL, SVBR or human readable text. Schematron (ISO/IEC 19757-3) is a rule-based validation language for making assertions about the presence or absence of patterns in XML. Constraints encoded using Schematron may be directly encoded within the resulting Application Schema or may be defined in an associated Schematron document.

10b-13 Dataset level metadata and integrity checks

The S-100 GML Profile does not explicitly contain any elements relating to dataset level metadata or integrity checks. S-100 Part 15 includes such provisions.

10b-14 Schema locations and namespaces

The GML profile and GML application schema for common elements are located at the IHO Geospatial Information (GI) Registry web site. Namespaces and versions are also defined on that site.

10b-15 Divergences from common GML practices

The GML profile (10b-8) and base schema (10b-11) diverge from common GML practice in the following items:

1. Interpretation of missing curve interpolation value (see clause 10b-10.1.2).
2. Geometry properties are defined individually instead of using substitution groups. There is no single property which functions as a spatial attribute in all features.

Page intentionally left blank

Appendix 10b-A
Application Schema
(informative)

10b-A-1

[To be added]

Page intentionally left blank

Appendix 10b-B
Use of Profile in GML Application Dataset
(informative)

10b-B-1 Introduction

This clause illustrates the use of the GML profile (10b-8) and base schema (10b-11) and a GML application schema (App. 10b-A) for an S-100-based data product and a GML dataset.

[To be added]

10b-B-2 Dataset structure in GML application schema

An example of the format of a GML dataset is shown in the Figure below. This dataset defines data objects as information objects, spatial objects or feature objects. It specifies the sequence of objects in the file as information objects first, followed by spatial objects, then features.

[To be added]

Page intentionally left blank

S-100 – Part 10c

HDF5 Data Model and File Format

Contents

10c-1 Scope

The Hierachical Data Format 5 (HDF5) HDF has been developed by the HDFgroup as a file format for the transfer of data that is used for imagery and gridded data. This Part is a profile of HDF5 and specifies an interchange format to facilitate the moving of files containing data records between computer systems. It defines a specific structure which can be used to transmit files containing data types and data structures conforming to the S-100 General Feature Model.

This Part specifies constraints and conventions that collectively specify the rules for S-100 HDF5 data formats. HDF5 features not required by S-100 HDF5 data are excluded. The scope of this Part is limited to the data format and does not include the application schema, nor does it include guidelines for how to develop Product Specifications or naming rules for features and attributes.

10c-2 Introduction

HDF5 uses an open source format. It allows users such as the IHO to collaborate with The HDF Group regarding functionality requirements and permits users' experience and knowledge to be incorporated into the HDF product when appropriate.

HDF5 is particularly good at dealing with data where complexity and scalability are important. Data of virtually any type or size can be stored in HDF5, including complex data structures and data types. HDF5 is portable, running on most operating systems and machines. HDF5 is scalable - it works well in high end computing environments, and can accommodate data objects of almost any size or multiplicity. It also can store large amounts of data efficiently - it has built-in compression. HDF5 is widely used in government, academia, and industry.

10c-3 Conformance

The S-100 HDF5 data format conforms to release 1.8.8 of HDF5.

10c-4 References

10c-4.1 Normative references

The HDF Group, November 2011*, HDF5 User's Guide Release 1.8.8*

The HDF Group, November 2011, *HDF5 Reference Manual 1.8.8*

ISO 8601:2004, *Data elements and interchange formats – Information interchange – Representation of dates and times*

ISO 19123, *Geographic information — Schema for coverage geometry and functions*

ISO 19129:2009, *Geographic information – Imagery, gridded and coverage data framework*

10c-4.2 Informative references

Gilbert, W., *A Cube-filling Hilbert Curve,* Mathematical Intelligencer 6(3), p.78, 1984

Goodchild, M. F. and Grandfield, A. W., *Optimizing Raster Storage: An Examination of Four Alternatives,* Proceedings Auto-Carto 6(1), pp. 400-407), Ottawa, 1983

Kidner, D.B., Higher-order interpolation of regular grid digital elevation models, International Journal of Remote Sensing, 24(14), July 2003, pp. 2981-2987. DOI: 10.1080/0143116031000086835

Kidner, D., Mark Dorey, M., & Smith, D., *What's the point? Interpolation and extrapolation with a regular grid DEM*, Proceedings of the 4th International Conference on GeoComputation, Fredericksburg, Virginia. URL: http://www.geocomputation.org/1999/082/gc_082.htm (retrieved 26 April 2018)

Laurini, R. and Thompson, D., *Fundamentals of Spatial Information Systems*, Academic Press, 1992

10c-5 HDF5 Specification

HDF5 implements a model for managing and storing data. The model includes an abstract data model and an abstract storage model (the data format), and libraries to implement the abstract model and to map the storage model to different storage mechanisms. The HDF5 library provides a programming interface to a concrete implementation of the abstract models. The library also implements a model of data transfer; that is, efficient movement of data from one stored representation to another stored representation. The Figure below illustrates the relationships between the models and implementations.

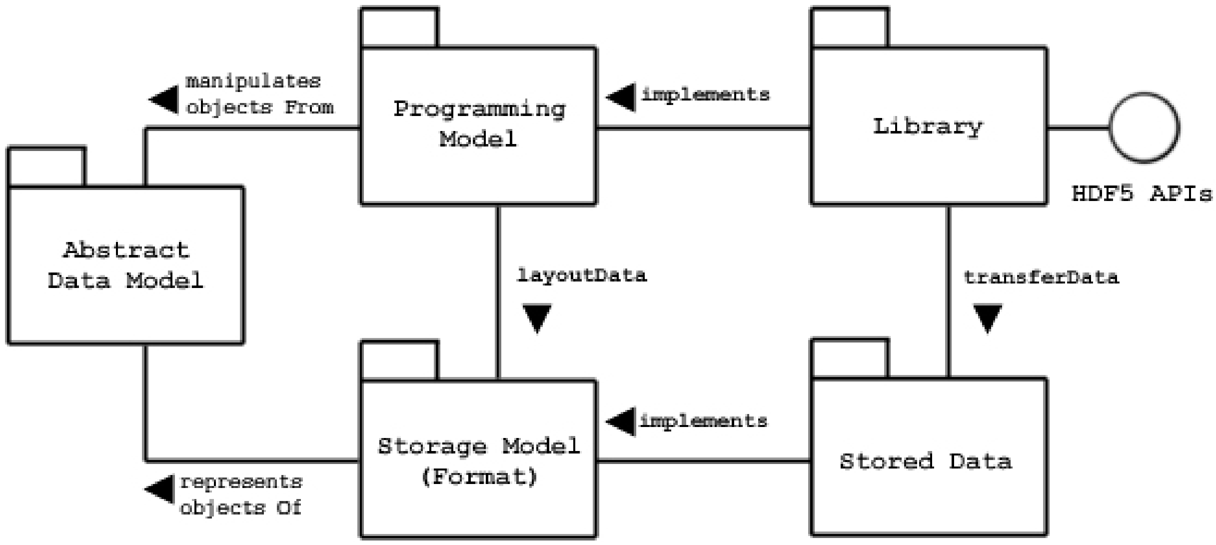


Figure 10c-1 – HDF5 models and implementation

The *Abstract Data Model* is a conceptual model of data, data types, and data organization. The abstract data model is independent of storage medium or programming environment. The *Storage Model* is a standard representation for the objects of the abstract data model. The *HDF5 File Format Specification* defines the storage model.

The *Programming Model* is a model of the computing environment and includes platforms from small single systems to large multiprocessors and clusters. The programming model manipulates (instantiates, populates, and retrieves) objects from the abstract data model.

The *Library* is the concrete implementation of the programming model. The Library exports the HDF5 APIs as its interface. In addition to implementing the objects of the abstract data model, the Library manages data transfers from one stored form to another. Data transfer examples include reading from disk to memory and writing from memory to disk.

Stored Data is the concrete implementation of the storage model. The storage model is mapped to several storage mechanisms including single disk files, multiple files (family of files), and memory representations.

The HDF5 Library is a C module that implements the programming model and abstract data model. The HDF5 Library calls the operating system or other storage management software (e.g., the MPI/IO Library) to store and retrieve persistent data. The HDF5 Library may also link to other software such as filters for compression. The HDF5 Library is linked to an application program which may be written in C, C++, Fortran, or Java. The application program implements problem specific algorithms and data structures and calls the HDF5 Library to store and retrieve data.

The HDF5 Library implements the objects of the HDF5 abstract data model. Some of these objects include groups, datasets, and attributes. An S-100 Product Specification maps the S-100 data structures to a hierarchy of HDF5 objects. Each S-100-based Product Specification will create a mapping best suited to its purposes.

The objects of the HDF5 abstract data model are mapped to the objects of the HDF5 storage model, and stored in a storage medium. The stored objects include header blocks, free lists, data blocks, B-trees, and other objects. Each group or dataset is stored as one or more header and data blocks.

10c-5.1 Abstract Data Model

The abstract data model (ADM) defines concepts for defining and describing complex data stored in files. The ADM is a very general model which is designed to conceptually cover many specific models. Many different kinds of data can be mapped to objects of the ADM, and therefore stored and retrieved using HDF5. The ADM is not, however, a model of any particular problem or application domain. Users need to map their data to the concepts of the ADM.

The key concepts include:

- *File* - a contiguous string of bytes in a computer store (memory, disk, etc), and the bytes represent zero or more objects of the model;
- *Group* - a collection of objects (including groups);
- *Dataset* - a multidimensional array of data elements with attributes and other metadata;
- *Dataspace* - a description of the dimensions of a multidimensional array;
- *Datatype* - a description of a specific class of data element including its storage layout as a pattern of bits;
- *Attribute* - a named data value associated with a group, dataset, or named datatype;
- *Property List* - a collection of parameters (some permanent and some transient) controlling options in the library;
- *Link* - the way objects are connected.

These key concepts are described in more detail below.

10c-5.1.1 File

Abstractly, an HDF5 file is a container for an organized collection of objects. The objects are groups, datasets, and other objects as defined below. The objects are organized as a rooted, directed graph. Every HDF5 file has at least one object, the root group. See the figure below. All objects are members of the root group or descendents of the root group.

HDF5 objects have a unique identity *within a single HDF5 file* and can be accessed only by its names within the hierarchy of the file. HDF5 objects in different files do not necessarily have unique identities, and it is not possible to access a permanent HDF5 object except through a file.

When the file is created, the *file creation properties* specify settings for the file. The file creation properties include version information and parameters of global data structures. When the file is opened, the *file access properties* specify settings for the current access to the file. File access properties include parameters for storage drivers and parameters for caching and garbage collection. The file creation properties are set permanently for the life of the file, and the file access properties can be changed by closing and reopening the file.

An HDF5 file can be “mounted” as part of another HDF5 file. This is analogous to Unix file system mounts. The root of the mounted file is attached to a group in the mounting file, and all the contents can be accessed as if the mounted file were part of the mounting file.

10c-5.1.2 Group

An HDF5 group is analogous to a file system directory. Abstractly, a group contains zero or more objects, and every object must be a member of at least one group. The root group is a special case; it may not be a member of any group.

Group membership is actually implemented via link objects. See the Figure below. A link object is owned by a group and points to a named object. Each link has a name, and each link points to exactly one object. Each named object has at least one and possibly many links to it.

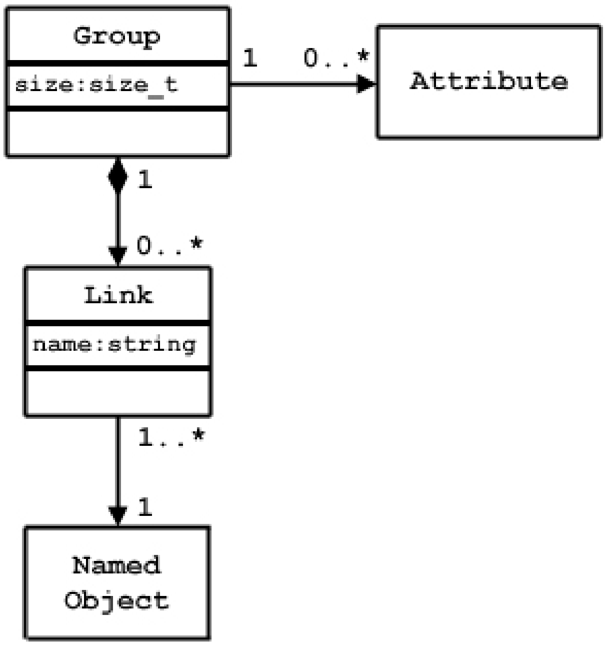


Figure 10c-2 - Group membership via link objects

There are three classes of named objects: group, dataset, and named datatype. See the Figure below. Each of these objects is the member of at least one group, and this means there is at least one link to it.

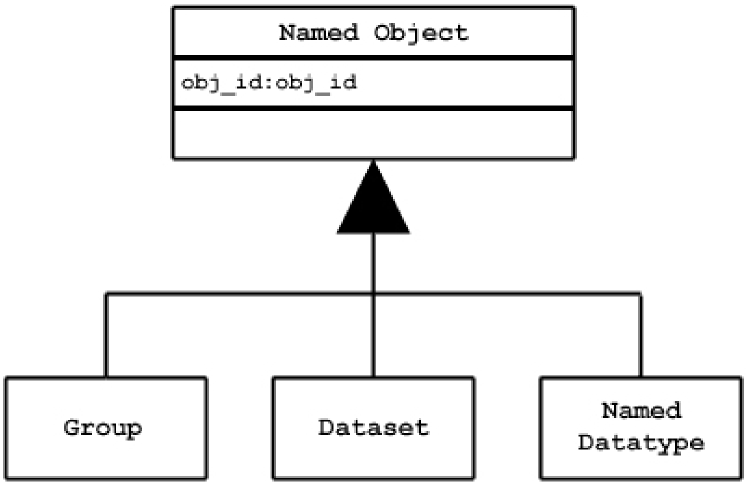


Figure 10c-3 - Classes of named objects

10c-5.1.3 Dataset

An HDF5 dataset is a multidimensional array of data elements. See the Figure below. The shape of the array (number of dimensions, size of each dimension) is described by the dataspace object.

A data element is a single unit of data which may be a number, a character, an array of numbers or characters, or a record of heterogeneous data elements. A data element is a set of bits. The layout of the bits is described by the datatype.

The dataspace and datatype are set when the dataset is created, and they cannot be changed for the life of the dataset. The dataset creation properties are set when the dataset is created. The dataset creation properties include the fill value and storage properties such as chunking and compression. These properties cannot be changed after the dataset is created.

The dataset object manages the storage and access to the data. While the data is conceptually a contiguous rectangular array, it is physically stored and transferred in different ways depending on the storage properties and the storage mechanism used. The actual storage may be a set of compressed chunks, and the access may be through different storage mechanisms and caches. The dataset maps between the conceptual array of elements and the actual stored data.

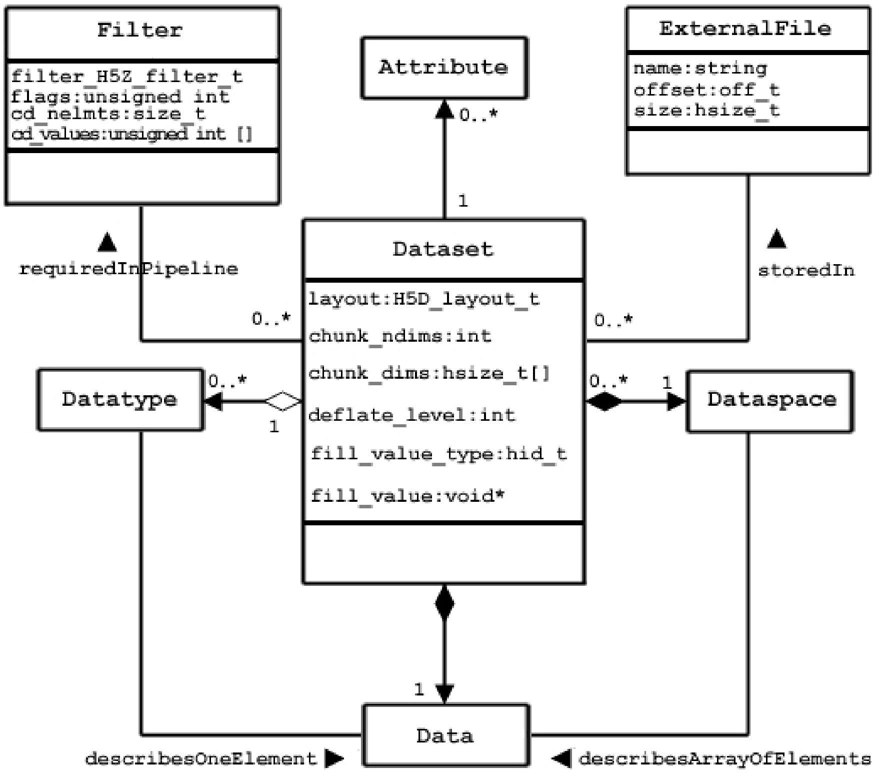


Figure 10c-4 - The dataset

The HDF5 concept of ‘dataset’ means an array, while the S-100 concept is defined as “an identifiable collection of data” (S-100 Annex A – Terms and Definitions) which is generally interpreted to mean a collection of instances of feature and/or information type.

This Part frequently uses the terms “data file” to mean a dataset in the S-100 sense and “HDF5 dataset” to mean a dataset in the HDF sense. Where these terms are not used, the sense should be apparent from the context.

10c-5.1.4 Dataspace

The HDF5 dataspace describes the layout of the elements of a multidimensional array. Conceptually, the array is a hyper-rectangle with one to 32 dimensions. HDF5 dataspaces can be extendable. Therefore, each dimension has a current size and a maximum size, and the maximum may be unlimited. The dataspace describes this hyper-rectangle: it is a list of dimensions with the current and maximum (or unlimited) sizes.

10c-5.1.5 DataType

The HDF5 datatype object describes the layout of a single data element. A data element is a single element of the array; it may be a single number, a character, an array of numbers or carriers, or other data. The datatype object describes the storage layout of this data.

Data types are categorized into 11 classes of datatype. Each class is interpreted according to a set of rules and has a specific set of properties to describe its storage. For instance, floating point numbers have exponent position and sizes which are interpreted according to appropriate standards for number representation. Thus, the datatype class tells what the element means, and the datatype describes how it is stored.

The Figure below shows the classification of datatypes. Atomic datatypes are indivisible. Each may be a single object; a number, a string, or some other objects. Composite datatypes are composed of multiple elements of atomic datatypes. In addition to the standard types, users can define additional datatypes such as a 24-bit integer or a 16-bit float.

A dataset or attribute has a single datatype object associated with it. See Figure 10c-4 above. The datatype object may be used in the definition of several objects, but by default, a copy of the datatype object will be private to the dataset.

Optionally, a datatype object can be stored in the HDF5 file. The datatype is linked into a group, and therefore given a name. A *named datatype* can be opened and used in any way that a datatype object can be used.

Not all the HDF5 datatypes have exact equivalents in the S-100 basic and derived datatypes defined in Part 1, clause 1-4.5.2 (Table 1-2). The correspondences between HDF5 and S-100 datatypes are given in Table 10c-2 later in this Part.

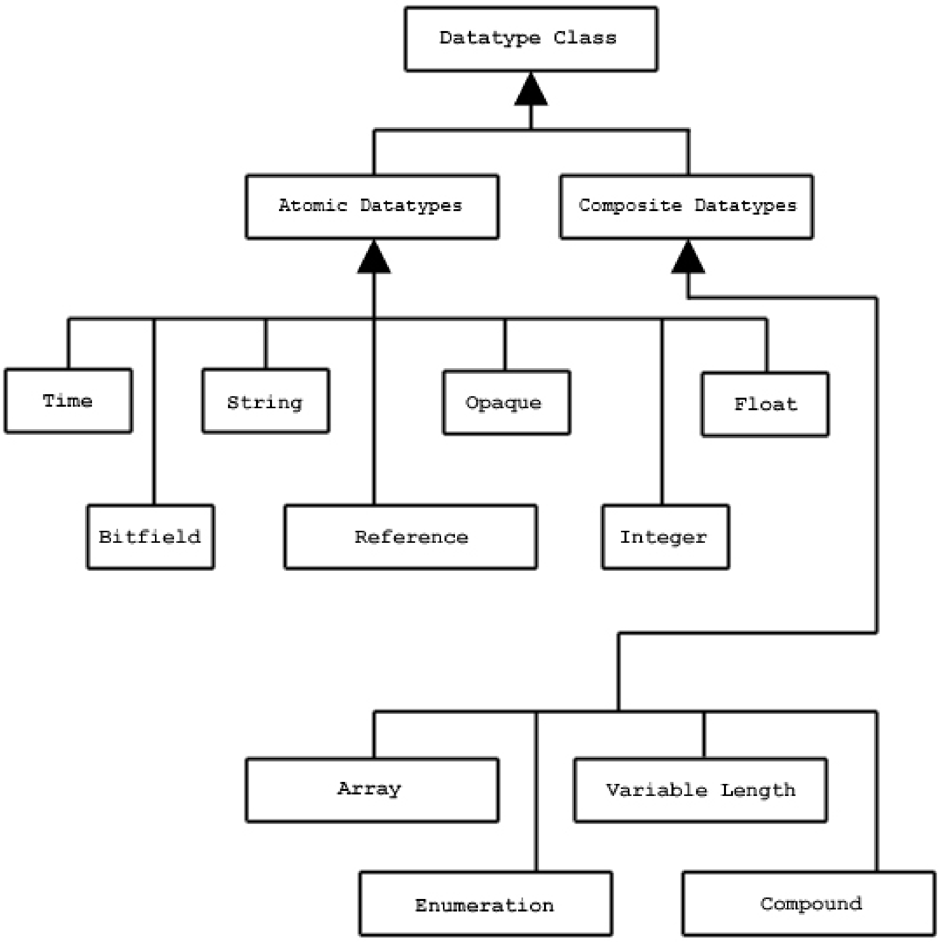


Figure 10c-5 - Datatype classifications

10c-5.1.6 Attribute

Any HDF5 named data object (group, dataset, or named datatype) may have zero or more user defined attributes. Attributes are used to document the object. The attributes of an object are stored with the object.

An HDF5 attribute has a name and data. The data portion is similar in structure to a dataset: a dataspace defines the layout of an array of data elements, and a datatype defines the storage layout and interpretation of the elements. See Figure 10c-6 below.

Attributes of data objects are in principle equivalent to thematic attributes but this edition of the HDF5 profile does not provide for vector feature or information type data in HDF5 files and therefore does not make use of vector object attributes. HDF5 attributes of groups, datasets, or named datatypes play the role of metadata.

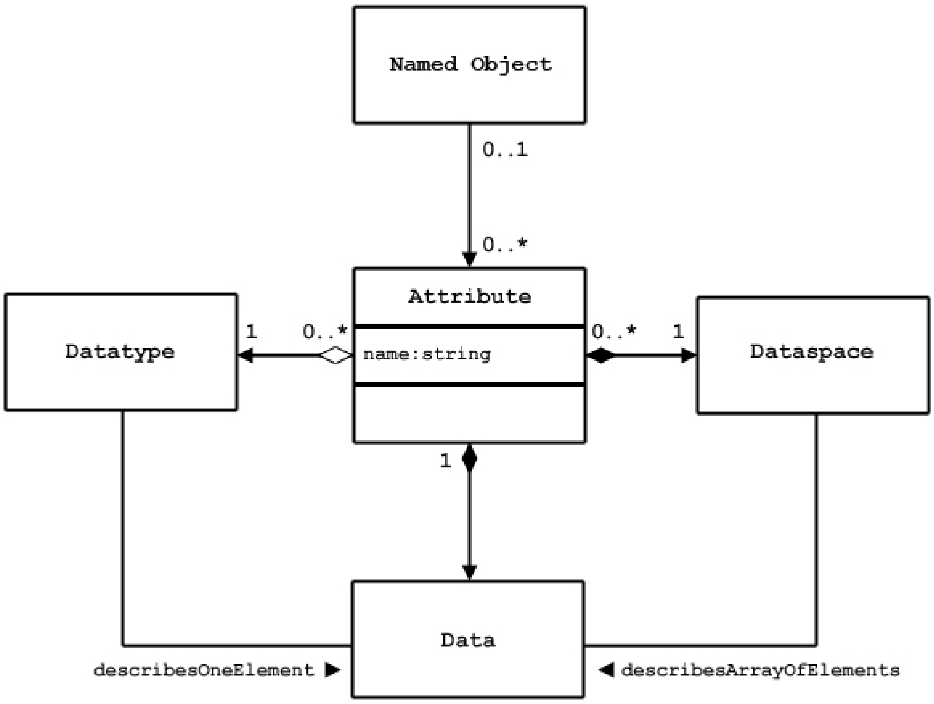


Figure 10c-6 - Attribute data elements

In fact, an attribute is very similar to a dataset with the following limitations:

- An attribute can only be accessed via the object;
- Attribute names are significant only within the object;
- An attribute should be a small object;
- The data of an attribute must be read or written in a single access (partial reading or writing is not allowed);
- Attributes do not have attributes.

Note that the value of an attribute can be an *object reference*. A shared attribute or an attribute that is a large array can be implemented as a reference to a dataset.

The name, dataspace, and datatype of an attribute are specified when it is created and cannot be changed over the life of the attribute. An attribute can be opened by name, by index, or by iterating through all the attributes of the object.

10c-5.1.7 Property List

HDF5 has a generic property list object. Each list is a collection of *name-value* pairs. Each class of property list has a specific set of properties. Each property has an implicit name, a datatype, and a value. A property list object is created and used in ways similar to the other objects of the HDF5 library.

Property Lists are attached to the object in the library, they can be used by any part of the library. Some properties are permanent (e.g., the chunking strategy for a dataset), others are transient (for example buffer sizes for data transfer). A common use of a Property List is to pass parameters from the calling program to a VFL driver or a module of the pipeline.

Property lists are conceptually similar to attributes. Property lists are information relevant to the behavior of the library while attributes are relevant to the user's data and application. Since the

Property List couples the data specification to an implementation use of HDF5 property lists in S-100 Product Specifications is discouraged.

10c-5.2 HDF5 Library and Programming Model

The HDF5 Library implements the HDF5 abstract data model and storage model. Two major objectives of the HDF5 products are to provide tools that can be used on as many computational platforms as possible (portability), and to provide a reasonably object-oriented data model and programming interface.

Refer to the HDF5 User's Guide Release 1.8.8 and the HDF5 Reference Manual 1.8.8 for more details on the HDF5 model implementation. S-100 Product Specifications must specify the HDF5 groups, datasets and attributes in context of the S-100 General Feature Model.

10c-5.3 Prohibited HDF5 constructs

Constructs which cannot be processed using the standard libraries of the HDF5 release specified in this Part must not be used. This means specifically that HDF5 constructs which require the use of a library for a later release than that specified in this Part must not be used.

10c-6 S-100 profile of HDF5

The S-100 profile of HDF5 restricts the HDF5 datatypes and constructs which can be used in S-100 HDF5 datasets; describes correspondences between S-100 and HDF5 datatypes and other constructs; and defines rules for how S-100 HDF5 datasets must be structured.

The S-100 HDF5 profile must apply to the kinds of information listed below – noting that the types are not all mutually exclusive, though most individual Product Specifications will use only a subset of possible combinations:

- data for one or more individual, fixed stations;
- regularly-gridded data;
- irregularly-gridded data;
- grids with variable cell sizes;
- ungeorectified gridded data (Part 8 clause 8-8.1.2);
- TIN data;
- moving platform (for example surface drifter) data;
- either static data or time series data (for any of the other kinds), with fixed or variable intervals;
- tiled and untiled coverages;
- multiple feature classes in the same datafile;
- multiple types of coverages in the same datafile.

The restrictions, correspondences, and rules are described in the following sections.

10c-7 Data types

Predefined HDF5 data types include Integer, Float, String, and Enumeration, but there are no HDF5 equivalents to the S-100 data types Boolean, S100_Codelist or S100_TruncatedDate. The latter types are mapped to the HDF5 constructs specified in the Table below. The S-100 data types Date, DateTime, and Time are mapped to HDF5 strings due to potential problems with portability across different processor architectures of HDF5 Time formats. In S-100 HDF5 data products, S-100 data types defined in Part 3 are mapped to equivalent HDF5 data types. These equivalences are summarized in Table 10c-1 below. HDF5 datatype classes not mentioned in this Table shall not be used.

Table 10c-1 – Equivalences between S-100 and HDF5 datatypes

S-100 Attribute Value Types	HDF5 Datatype Class	Constraint on HDF5 datatype
real	Float	32 or 64-bit floating point
integer	Integer	1, 2, or 4-byte signed and unsigned integers
text (CharacterString in S-100 metadata)	String	variable-length string
enumeration	Enumeration	Numeric codes must be 1 or 2-byte unsigned integers, range [1, 2^8 – 1] or [1, 2^{16} - 1]
date	(Character) String, length=8	Date format according to Table 1-2 (Part 1); that is, complete representation, basic format, as specified by ISO 8601
time	(Character) Variable-length string	Time format according to Table 1-2 (Part 1); that is, complete representation, basic format as specified by ISO 8601. UTC indicated by “Z” suffix; local time by absence of suffix. The zone offset format is also permitted); for example, 123000+0100
dateTime	(Character) (variable length string)	Date-time format as specified by ISO 8601. EXAMPLES: 19850412T101530Z 19850412T101530-0500
boolean	(Integer)	1-byte unsigned, Values: 1 (TRUE); 0 (FALSE)
S100_Codelist	Compound (Enumeration, variable-length string)	Exactly one of the components is allowed; the other must be the numeric value 0 or the empty (0-length) string according to its data type
URI, URL, URN	String (variable-length)	Format specified in RFC 3986 (URI, URL) or RFC 2141 (URN)
S100_TruncatedDate	String, length=8	Format as in Part 1 Table 1-2
value record (Part 8)	Compound	Datatypes of components must be according to value attribute types in the Application Schema. The “value record” corresponds to the value(s) record in Part 8 Figures 8-22, 8-23, 8-24 and 8-25
external object reference	String	Format: extObjRef:<fileName>:<recordIdentifier> where <fileName> is the base name of the ISO 8211 or GML file, and <recordIdentifier> is the record identifier of the vector object record within that file. The extension part of the file name is not used. The record identifier is the gml:id for GML datasets, or the record identification number (RCID) for ISO 8211 datasets. The file must be present in the same exchange set

10c-8 Naming conventions

Names of HDF5 elements (datasets, objects, etc) that encode data elements in the Application Schema (that is, feature classes, attributes, roles, enumerations, codelists, etc) must conform to the names in the Application Schema (since there is 1/1 mapping from the Application Schema to the Feature Catalogue, this also amounts to requiring the same conformance to the Feature Catalogue). ‘Names’ used must be the camel case names. Other sections in this Part indicate where the names from the Application Schema (or equivalently, the Feature Catalogue) are used.

Elements in embedded ("carrier") metadata and positioning information which correspond to attributes in Parts 4a-4c must also conform to the corresponding camel case names in Parts 4a-4c & 8.

Elements which do not have a direct correspondence may have names that are unique to the HDF5 format (the differences being intended to simplify the abstractions in ISO 19123 and S-100 Parts 4, 4b, and 8, and shorten fields which are deeply nested within the XML schemas).

The names 'latitude' and 'longitude' must be used for geographic coordinate axes when they are appropriate, in preference to 'X' and 'Y', which should be used only when latitude/longitude are inappropriate.

The correpondences between the carrier metadata elements in this profile and Parts 4-4c and Part 8 are specified later in this document.

Names in non-embedded metadata and catalogue files in exchange sets are treated as for vector product Product Specifications – that is, they must conform to the standard S-100 metadata and exchange catalogue schemas.

An HDF5 group which corresponds to a schema element already named in S-100 or in the Product Specification must be given the same name as that element, using the camel-case code if specified. For example, if a time series product specifies names for data collections at time points, those names should be used as the group names if the collection is encoded as a group. (Product Specification developers must take care to specify collection names which conform to the allowed HDF5 syntax.)

Numeric suffixes preceded by the underscore character (that is, the suffix 'NNN') may be added to distinguish groups which would otherwise have the same names (for example, data groups at different time points).

The following group names are reserved for the uses specified:

Table 10c-2 – Reserved group names

Positioning	Discrete positioning information of all kinds and dimensions. The type of positioning data is indicated by a group attribute or attributes. Includes compressed or compact encodings. Does not include positioning which can be completely specified by grid or coverage parameters alone (such parameters are encoded in attributes attached to the root group). Specifications which require non-uniform positioning (for example, second-order algebraic formulae) must be treated as ungeorectified grids.
Group_F	Feature specification information. For example, feature and attribute names, codes, types, multiplicities, roles, etc. Also includes format metadata specific to the HDF5 format.
Group_IDX	Indexes, if encoded in an HDF5 group. Includes indexes to sparse arrays.
Group_TL	Tiling information, if encoded in a group.
Group_nnn	Data for one member of a series; for example, at a time point in a time series, or for different stations. "n" means any digit from 0 to 9. Numbering must use 3 digits, 001-999.

10c-9 Structure of data product

10c-9.1 General structure

An S-100 HDF5 file is structured to consist of Groups, each of which may contain other Groups, Attributes and (HDF) Datasets. Groups are containers for different types of information (meaning data values, position information, metadata, or ancillary information). HDF datasets are designed to hold large amounts of numerical data and may be used to hold the coverage data values. Attributes are designed to hold single-valued information which apply to Groups or Datasets and may be used to hold certain types of metadata.

The following groups are contained within the root group. (The nesting levels in the list below correspond to the nesting levels in the HDF5 file.)

1) Feature information group.

2) Feature container groups – each acts as a container for individual instances of a feature class. Its attributes encode any feature-class-level metadata.
   a) Feature instance groups – each acts as a container for the positioning, tile, indexes, and data groups pertaining to a single feature instance. Its attribute encode any instance-level metadata
      i) Tiling information group (conditional, only if values are stored as tiles).
      ii) Indexes group (conditional, only if indexes to data are required).
      iii) Positioning group (conditional, only if positions are not computable from metadata).
      iv) Data values group(s). Only time series data will have more than one value group.

Note that the order in which groups and datasets are stored within the datafile may not be the same as the order in which they are created.

The basic structure of an S-100 HDF5 file is depicted in the Figure below. 'F' is the number of feature classes defined in the Product Specification. It is not a requirement that every data file contain instances of all feature classes. There is one values group for each time point in the time series[1] (datasets which are not time series will have only a single values group in each feature instance group).

The FeatureContainer and Positioning groups are abstract classes because their attributes and content depend on the type of coverage.

A more detailed diagram is included later in this Part.

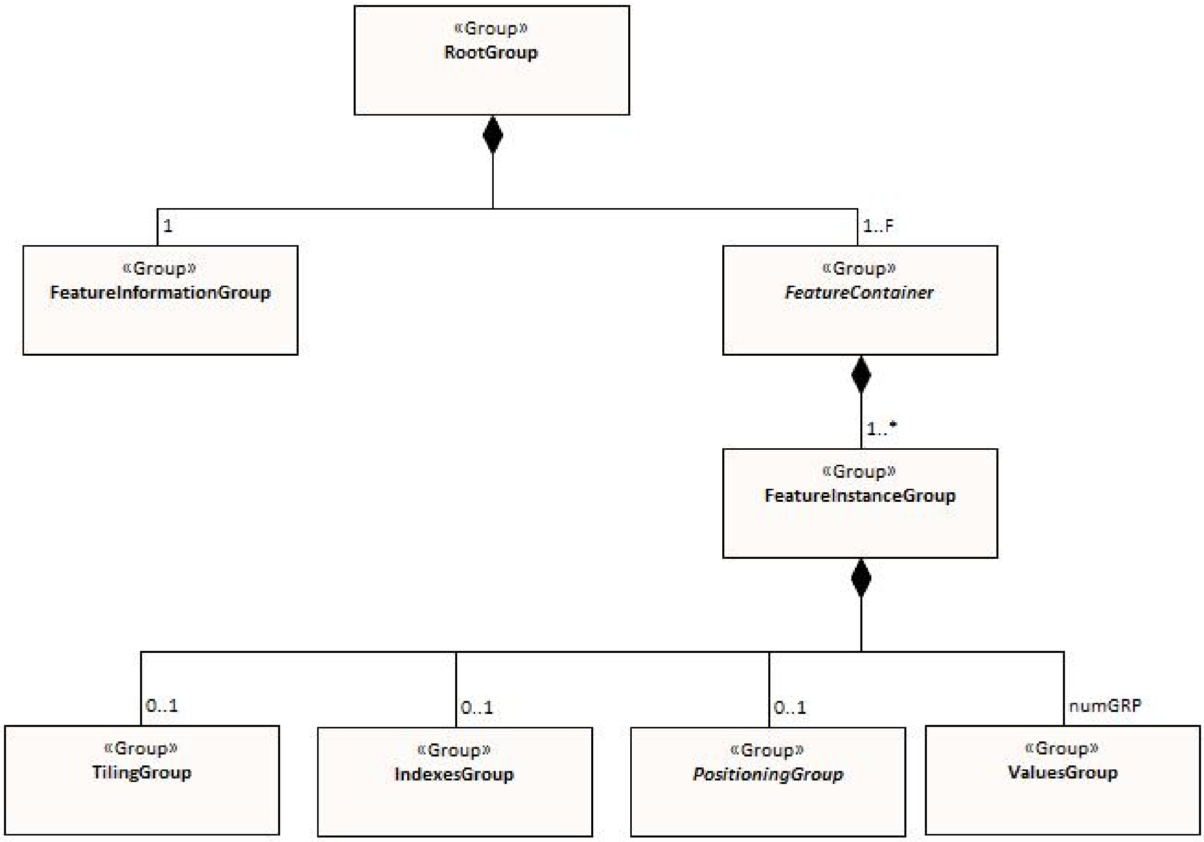


Figure 10c-7 - Basic structure of S-100 HDF5 file

[1] Except for moving station data and fixed station (stationwise) data. The use of value groups for each coverage type is described later in this Part.

10c-9.2 Metadata

Metadata is defined at different levels in the logical structure, so that metadata at the root group applies to all the features in the file, metadata at the feature container level applies to all instances of that feature class, and metadata at the instance level applies only to that particular feature instance.

10c-9.2.1 Discovery metadata

Full discovery metadata is encoded in an external discovery metadata file, as specified in Parts 4a (Metadata) and 4b (Metadata for Imagery and Gridded Data). See clause 10c-12 for naming conventions.

10c-9.2.2 Carrier (embedded) metadata

Carrier metadata is metadata that is encoded within the HDF5 file. It is divided into general, type, and instance metadata, depending on whether it pertains to the HDF5 file as a whole, describes the structure and attributes of data object classes, or provides parameters needed to read instances of data object classes. Metadata is encoded in the following places:

- General metadata, defined as general parameters that apply to the file as a whole. General metadata consists of parameters that apply to all information in the data file, such as dates of issue, datum information, and overall spatial extent (bounding box). This includes the essential general elements for processing and cell location (the rest of the essential information is encoded with the feature instance). This metadata is encoded as attributes of the root group;
- Type, or feature, metadata, defined as specific characteristics which describes data object classes in the file (for example, pertains to specific features and attributes) and which will therefore be different for each feature class. This metadata is used for feature and attribute specification information (corresponding to entries in the feature catalogue). This type information is analogous to the feature catalogue described in Part 5, but may contain only extracts from the Feature Catalogue as well as add format-specific parameters relevant only to HDF5 encodings. The Type Metadata is encoded as content (HDF5 datasets) in the feature information group and as attributes of each feature container group. The feature information group (Group_F) is also the future intended container for information from the exchange set catalogue or about support files, if it is necessary to include that within the HDF5 file and it is not applicable to the file as a whole;
- Instance metadata, defined as parameters for each feature class in the application schema. This includes parameters that are needed to read the information in the data product even if external metadata files are unavailable, including coverage-specific spatial parameters (extent, grid parameters). This metadata may include parameters that have significance only in the context of the specific coverage spatial type(s) permitted for the feature class in the application schema. This metadata is encoded as attributes of the instances within each feature container group.

Additional information describing the data is contained in the values group, as attributes that apply to the values dataset in each values group. The data may be a time point or station information such as station name; and the time series characteristics such as time interval, number of values and start and end times.

10c-9.2.3 Extended metadata

Extended metadata elements defined in the Product Specification are encoded as either or both of:

- Additional attributes of the root of feature container group, depending on whether they are considered necessary for processing and pertain to the datafile as a whole or to feature instances. An example is provided later in this Part (Table 10c-7). (Note that any extended metadata that is essential for processing implies product-specific modules in implementations.);
- Extended metadata in the external XML files encoding the discovery metadata or exchange catalogue, if they are considered discovery metadata.

Data products may also define vector feature metadata; for example, quality meta-features with vector geometry. Vector features are not encoded within the HDF5 file but in a separate file conforming to S-

100 Part 10a or Part 10b. If vector meta-features are present, a reference to the separate file must be included in carrier metadata by naming the file in the *metaFeatures* attribute (see clause 10c-9.4).

10c-9.3 Generalized dimensions and storage of coordinates and data

This section provides an overview of the general approach to representing positioning information and storing data in S-100 HDF5 datasets. The basic approach is to minimize the variety of data structures used for storing data records. This profile stores data in one of two ways:

1) A multi-dimensional data array, of rank and dimensions corresponding exactly to the shape of the grid. This is used only for regular grids. In order to reduce space requirements, the coordinates of grid points are not explicitly stored because they can be computed from grid parameters;
2) One-dimensional arrays of data and grid coordinates, accompanied by meta-information describing the shape of the grid. This is also used for multipoint data (where there is no actual grid).

The key idea at the core of the structure is this: the organization of the data is logically the same for each of the various types of data, but the information itself will be interpreted differently depending on the type of spatial representation, which is indicated by the metadata attribute dataCodingFormat (defined in Table 10c-10 and clause 10c-4).

For regularly-gridded data, the positioning information is not stored in the form of explicit coordinates because the grid metadata (extent and grid cell spacing information) suffices to specify the coordinates of each grid point. For example, for 2-D grids the value arrays are two dimensional, with dimensions specified by the attributes numPointsLongitudinal and numPointsLatitudinal. By knowing the grid origin and the grid spacings, the position of every point in the grid can be computed by simple formulae.

For non-regularly gridded data only, there is additional positioning information. The nature of the positioning information depends on the data type:

- For fixed stations, fixed stations (stationwise) and moving platform data, the positioning information is stored as explicit coordinates, in one-dimensional arrays of size numPOS of compound elements. The components of the compound element correspond to the coordinate axes; for example, latitude, longitude, z-coordinate, time, etc. The sequence of points corresponds either to the positions of fixed stations or sequential positions of moving platforms, as appropriate.
- For ungeorectified grids, the positioning information is also stored as explicit coordinates in one-dimensional arrays of size numPOS of compound elements that contain the coordinates (as defined above).
- For irregular grids, the positioning information is stored as one-dimensional arrays of size numPOS of compound elements containing information about the location of populated cells. Coordinate values for each grid point are not explicitly stored. In addition, the tiling group may be populated with tiles whose spatial union exactly covers the grid. The sequence of cell location arrays must conform to the sequencingRule metadata attribute in the feature container group (clause 10c-9.6). An optional tile index component (index into the tiles array – see clause 10c-9.7) may be added to by a Product Specification for faster retrieval. If used, the tile index component must be named 'tileIndex' and be of 'integer' datatype. This format is intended for grids of irregular shapes based on uniform rectangular cells.
- For grids with variable cell sizes, the positioning information is stored as two one-dimensional arrays of size numPOS of compound elements, one array containing information about cell location (as for irregular grids) and the other about cell sizes. Coordinate values for each grid point are not explicitly stored. The actual cell size is described in terms of aggregations of a unit cell size. The format assumes that the varying cells are aligned with the grid and that cell sizes are multiples of unit cell size in each dimension.
- For TIN data, the positioning information is stored as one-dimensional arrays of size numPOS encoding the vertex locations (using the same type of compound elements as for ungeorectified grids above) plus a Triangles array encoding references to the vertices of the triangle and references to adjacent triangles.

For irregular grids and variable cell size, the auxiliary arrays describing cell locations and sizes are stored in the 'values' group rather than the positioning group (this allows for different aggregations of cells at different time points in the variable cell size format). The storage of data and coordinate values is summarized in the Table below. ('D' is the number of dimensions of the coverage.)

The HDF datasets storing coordinates and values are designed so as to use uniform data storage structures across different coverage types as well as reduce the total data volume. These criteria resulted in storing the additional information needed by some coverage types separately (e.g., cell location and size information for irregular and variable cell size grids).

Table 10c-3 – Summary of storage strategies for coordinates and data values

Coverage type	Coordinate values	Data values
Regular grid	Not explicitly stored Computable from metadata	D-dimensional array of value tuples
Irregular grid	Not explicitly stored Computable from metadata	1-d array of value tuples + information about location of cells
Variable cell size grid	Not explicitly stored Computable from metadata	1-d array of value tuples + information about cell size and location
Fixed stations, fixed stations (stationwise), ungeorectified grid, moving platform	1-d array of coordinate tuples	1-d array of value tuples
TIN	1-d array of coordinate tuples + triangle information	1-d array of value tuples

Data Groups are separate groups containing the data values, which are stored in arrays corresponding to the positioning information. For coverage types where positioning information is not explicitly stored (N-dimensional regular grids), data is stored in N-dimensional arrays of rank corresponding to the grid dimensions (for example, for 2-D data, 2-D arrays of size numROWS by numCOLS).

For time series data, multiple data groups are present. The total number of data Groups is numGRP. The meaning of numGRP for each type of spatial representation is specified in Table 10c-4 below. The format allows for time series data for all representations.

Positions in coordinate systems with more than 2 coordinate axes are encoded using correspondingly more dimensions. For example, for 3-dimensional data, the vertical dimension is used as a third dimension.

For processing efficiency, this profile recommends limiting the number of dimensions to no more than four (space and time), but higher dimensionality may be used if required for the data product.

The variables that determine the array sizes (numROWS, numCOLS, numPOS, and numGRP) are different, depending upon which coding format is used. They are given in Table 10c-4.

Table 10c-4 – Array dimensions for different types of coverages

Coding Format	Data Type	Positioning	Data Values			
		numPOS	numCOLS	numROWS	numZ (3-d only)	numGRP
1	Fixed Stations	numberOfStations	1	numberOfStations	1	numberOfTimes
2	Regular Grid	(not used)	numPointsLongitudinal	numPointsLatitudinal	numPointsVertical	numberOfTimes
3	Ungeorectified Grid	numberOfNodes	1	numberOfNodes	1	numberOfTimes
4	Moving Platform	numberOfTimes	1	numberOfTimes	1	1
5	Irregular Grid	numberOfNodes	1	numberOfNodes	1	numberOfTimes
6	Variable cell size	numberOfNodes	1	numberOfNodes	1	numberOfTimes
7	TIN	numberOfNodes	1	numberOfNodes	1	numberOfTimes
8	Fixed Stations (Stationwise)	numberOfStations	1	numberOfTimes	1	numberOfStations
9	Feature oriented Regular Grid	(not used)	numPointsLongitudinal	numPointsLatitudinal	1	numberOfTimes

Note that numROWS, numCOLS, numZ, and numPOS are not explicitly encoded in the HDF5 file. This specification uses them only to indicate array dimensions for implementation purposes. It is the number of stations, nodes, points, etc that are encoded as attributes of feature instances (clause 10c-9.7).

The name of each data Group begins with the characters 'Group_nnn', where n is numbered from 1 to numGRP. A maximum of 999 data groups are allowed. The length of the data group name is 9.

For all data types, the logical product structure in HDF5 consists of (a) a metadata block, which is followed by (b) the feature information group, then (c) one or more data container groups, each of which contains one or more feature instance groups, which in turn contain tiling, indexing, positioning and data groups as described in clause 10c-9.1. The tiling, indexing, and positioning groups are conditionally required depending on the type of data, indicated by an HDF5 attribute that specifies the coding format.

The physical layout of the file may not be the same as its logical data structure, however the HDF5 API allows implementers to access information using the logical data structure.

The following sections describe the content and attributes of each group.

10c-9.4 Root group

The root group acts as a container for the other groups. The carrier metadata (Table 10c-6) is contained as attributes in the root group. The carrier metadata consists of the data and parameters (a) needed to read and interpret the information in the product even if external metadata files are unavailable, and, mostly, (b) are not included elsewhere in the metadata.

Page intentionally left blank

Table 10c-5 – Root group

Group	HDF5 Category	Name	Data Type	Data Space / Remarks
/ *(root)*	Attributes	(Carrier metadata attributes)	Integer, Float, Enumeration, or String	(none) Described in Table 10c-6
	Group	Group_F		Feature information group (see clause 10c-9.6)
	Group(s)	(featureCode)		Feature container group – one group for each feature type in the data product. The name is the feature code, which is given in Group_F. See clause 10c-9.6 for structure and attributes
		HDF5 Category / **Name**		
		Group(s) / (featureCode).N		Feature instance group(s) – one for each instance of the feature. See Section 10c-9.7 for structure and attributes
		HDF5 Category / **Name**		
		Group (optional) / Group_TL		Tiling information, only if product uses tiles. See Section 10c-9.8
		Group (optional) / Group_IDX		Spatial index information, only if product uses spatial indexes See Section 10c-9.9
		Group / Positioning		Positioning information – 2D or 3D. Not required for dataCodingFormat = 2 (Regular grid) and dataCodingFormat = 9 (Feature oriented Regular Grid) See Section 10c-9.10
		Group(s) / Group_NNN		Static data – only 1 values group Time series data – 000 to 999 groups See Section 10c-9.11

The common (core) metadata elements are specified as attributes of the root group, as listed in Table 10c-6. The root group contains only a subset of the elements of minimum metadata specified in Parts 4a and 4b. The external XML metadata file is required to contain all the mandatory metadata elements.

Table 10c-6 – Embedded metadata (carrier metadata) in root group

No	Name	Camel Case	Mult	Data Type	Remarks and/or Units
1	Product Specification number and version	productSpecification	1	String	For example[2], 'INT.IHO.S-NNN.X.X', with Xs representing the version number. "NNN" and "X" do not imply length restrictions Corresponds to combination of S100_ProductSpecification name and number fields
2	Time of data product issue	issueTime	0..1	String (Time format)	Must be consistent with issueTime in discovery metadata
3	Issue date	issueDate	1	String (Date format)	Must be consistent with issueDate in discovery metadata
4	Horizontal CRS	horizontalCRS	1	Integer	EPSG code or -1 if user defined
5	Name of the horizontal CRS	nameOfHorizontalCRS	0..1	String	Mandatory if horizontalCRS = -1
6	Type of the horizontal CRS	typeOfHorizontalCRS	0..1	Enumeration	Mandatory if horizontalCRS = -1 See Table 10c-21
7	Horizontal coordinate system	horizontalCS	0..1	Integer	Mandatory if horizontalCRS = -1 Allowed values if typeOfHorizontalCRS = 1 (Geodetic CRS 2D): • 6422 (Lat, Lon – degree) Allowed values if typeOfHorizontalCRS = 2 (Projected CRS): • 4400 (Easting, Northing – metres) • 4500 (Northing, Easting – metres)
8	Horizontal datum	horizontalDatum	0..1	Integer	Mandatory if horizontalCRS = -1 EPSG code or -1 if user defined
9	Name of horizontal datum	nameOfHorizontalDatum	0..1	String	Mandatory if horizontalDatum = -1
10	Prime meridian	primeMeridian	0..1	Integer	Mandatory if horizontalDatum = -1; EPSG Code
11	Spheroid	spheroid	0..1	Integer	Mandatory if horizontalDatum = -1; EPSG Code
12	Projection method	projectionMethod	0..1	Integer	Mandatory if typeOfHorizontalCRS = 2; EPSG Code, see Table 10c-24
13	Projection parameter 1	projectionParameter1	0..1	Float	Only if projectionMethod is used. See Table 10c-24
14	Projection parameter 2	projectionParameter2	0..1	Float	Only if projectionMethod is used. See Table 10c-24
15	Projection parameter 3	projectionParameter3	0..1	Float	Only if projectionMethod is used. See Table 10c-24
16	Projection parameter 4	projectionParameter4	0..1	Float	Only if projectionMethod is used. See Table 10c-24

[2] To be replaced by a common format used in all S-100 based products, after that is finalized.

17	Projection parameter 5	projectionParameter5	0..1	Float	Only if projectionMethod is used. See Table 10c-24
18	False northing	falseNorthing	0..1	Float	Only if projectionMethod is used. To be applied to the coordinates at axis Northing. [m]
19	False easting	falseEasting	0..1	Float	Only if projectionMethod is used. To be applied to the coordinates at axis Easting. [m]
20	Epoch of realization	epoch	0..1	String	Code denoting the epoch of the geodetic datum used by the CRS. For example, G1762 for the 2013-10-16 realization of the geodetic datum for WGS84
21a	Bounding box	westBoundLongitude	1	Float	Ref. dataCoverage.boundingBox > EX_GeographicBoundingBox Each of the components of the bounding box is encoded as a separate attribute The minimum/maximum latitude and longitude of the data is based on a Geographic CRS that uses the same horizontal datum as the CRS defined for the data set which might be projected. The unit must be degrees.
21b		eastBoundLongitude	1	Float	
21c		southBoundLatitude	1	Float	
21d		northBoundLatitude	1	Float	
22	Geographic location of the resource (by description)	geographicIdentifier	0..1	String	EX_Extent > EX_GeographicDescription.geographicIdentifier > MD_Identifier.code
23	Metadata	metadata	1	String	MD_Metadata.fileIdentifier Name of XML metadata file (clause 10c-12). Ref. S-100 Part 8
24	Vertical coordinate system	verticalCS	0..1	Integer	EPSG Code; Allowed Values • 6498 (Depth – Metres – Orientation down) • 6499 (Height – Metres – Orientation up)
25	Vertical coordinate base	verticalCoordinateBase	0..1	Enumeration	See Table 10c-22
26	Vertical datum reference	verticalDatumReference	0..1	Enumeration	Only if verticalCoordinateBase = 2 See Table 10c-23
27	Vertical datum	verticalDatum	0..1	Integer	Only if verticalCoordinateBase = 2 If verticalDatumReference = 1 this is a value from S100_VerticalAndSoundingDatum If verticalDatumReference = 2 this is an EPSG code for vertical datum
28	Meta features	metaFeatures	0..1	String	Name of 8211 or GML file containing meta-features GML files must have extension .GML or .gml; ISO 8211 files must have extension .NNN where N is any digit

NOTES:

1) If the CRS is user defined only the following coordinate systems are supported:
   Geodetic CS (Latitude, Longitude) – Degrees; and

Cartesian CS (Northing, Easting or Easting, Northing) – Metres.

2) For the horizontal Datum all EPSG predefined Datum are allowed or any combination of predefined Prime Meridians or predefined Spheroids.
3) The projection methods are limited to those given in Table 10c-24.
4) If the horizontal CRS is defined by the EPSG code, the defined CRS should not use any other elements than the one allowed for user defined CRSs; (for example, no projection method that is not in the Table).
5) The bounding box is the data set bounding box; the coverage data feature instances may or may not cover the entire bounding box. If there is only a single coverage feature, its extent may or may not be the same as the data set.
6) The core attributes correspond to metadata attributes in S100_DatasetDiscoveryMetadata (Part 4a) or the imagery/gridded/coverage data attributes in Part 8. The correspondences are given in the Remarks column.
7) Vertical datum is optional since it is not applicable to some types of depth referencing as used in some data products; for example, Surface Currents.

Product Specifications which need additional metadata attributes may include them as additional attributes, defined in the Product Specification. The additional attributes must be defined in the same way as Table 10c-6 – specifically, they must have a camel-case name beginning with a lower-case letter, multiplicity either 0..1 (optional) or 1 (mandatory) and be one of the allowed types listed in Table 10c-1. In addition, restrictions or additional conditions can be added for core carrier metadata attributes. The data types of common carrier metadata attributes cannot be changed, but the range of allowed values may be restricted or optional attributes made mandatory or conditionally mandatory.

EXAMPLE: The Table below shows how a Product Specification might define an additional attribute (Vertical reference), introduce a conditional test for a core metadata attribute (Vertical datum reference), and make an optional metadata attribute mandatory (Time of data product issue).

Table 10c-7 – Example of extended metadata attribute and additional conditions on core metadata attributes

No	Name	Camel Case	Mult	Data Type	Remarks and/or Units
Additional carrier metadata					
11	Vertical reference	depthTypeIndex	1	Enumeration	1: Layer average 2: Sea surface 3: Vertical datum (see verticalDatum) 4: Sea bottom
Additional restrictions or conditions on core carrier metadata					
2	Time of data product issue	issueTime	1	String (Time format)	
9	Vertical datum reference	verticalDatum	0..1	Enumeration	Required if and only if depthTypeIndex=3

How the Product Specification describes core and extended metadata attributes is left to the specification writers, but specifications should distinguish core attributes from extended attributes as well as clearly indicating any additional restrictions or conditions on core attributes. The ISO format for specifying metadata extensions (Part 4a clause 4a-5.7.5) may be used.

10c-9.5 Feature information group

The feature information group contains the specifications of feature classes and their attributes. The components of the feature information group are described in the Table below.

Table 10c-8 – Components of feature information group

Group	HDF5 Category	Name	Data Type or HDF Category	Data Space
/Group_F	Dataset	featureCode	String (variable length)	Array (1-d): i=0, F-1 Values = codes of feature classes (F is the number of feature classes in the application schema.)
	Dataset(s) (feature information datasets - one for each feature in the featureCode array)	<featureCode> For example: SurfaceCurrent, WaterLevel	Array of Compound (String X 8)	Array (1-d): i=0, *NA_F-1 (NA_F = number of attributes of feature named by <featureCode>).* Components of the compound type: code: camel case code of attribute as in feature catalogue name: long name as in feature catalogue uom.name: units (uom>name from S-100 feature catalogue) fillValue: fill value (integer or float value, string representation) datatype: HDF5 data type, as returned by H5Tget_class() function lower: lower bound on value of attribute upper: upper bound on attribute value closure: type of closure The “code” and “datatype” components encode the rangeType attribute of the coverage features in Part 8 “lower”, “upper”, and “closure” encode any constraints on attribute values as encoded in the feature catalogue (see “S100_FC_SimpleAttribute>constraints” in Part 5 and S100_NumericRange in Part 1)

Notes:

1) Land mask or unknown values are represented by the attribute’s *fillValue*.

Page intentionally left blank

All the numeric values in the feature description dataset are string representations of numeric values; for example, “-9999.0” not the float value -9999.0. Applications are expected to parse the strings to obtain the numeric value. Inapplicable entries are represented by null values or the empty (0-length) string.

An entry in Group_F is required for every feature type that is used in the HDF5 data file. This means that:

- The **featureCode** array must include each feature type for which there is a feature instance somewhere in the current physical file.
- There must be a feature description dataset for each feature type named in the **featureCode** array.
- Each feature description dataset must list all the attributes of the feature type (both direct and inherited) as specified in the Feature Catalogue.

Note that the above requirements do not mandate entries in Group_F for feature types which are defined in the XML feature catalogue but for which there are no instances in the current data file.

The number of attributes for each feature type (NA_F in Table 10c-8) is not explicitly specified but can be determined using HDF5 API to determine the number of rows in each feature description dataset.

The Figure below depicts Group_F for a hypothetical product with two feature types, *SurfaceCurrent* and *WaterLevel*. The two features are named (using the camel case codes from the feature catalogue) in the dataset **featureCode**. The feature description datasets **SurfaceCurrent** and **WaterLevel** describe the attributes of each feature type. The feature description datasets are given the same names as the values in the **featureCode** dataset, which are the camel case codes of the features from the XML feature catalogue. Each feature description dataset is an array of compound type elements, whose components are the 8 components specified in Table 10c-8.

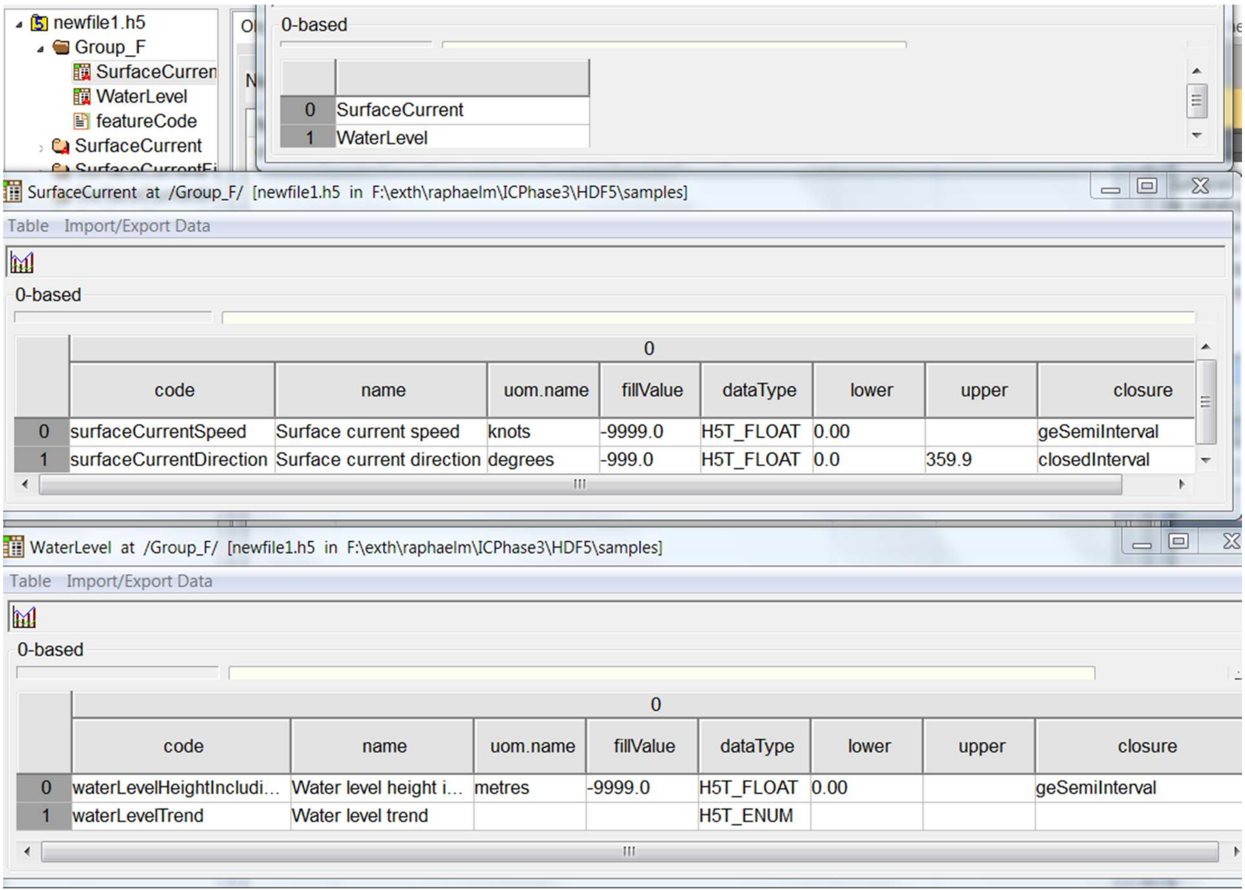


Figure 10c-8 – Example of Group_F

10c-9.6 Feature container group

The feature container groups contain the coordinates and values for all instances of a single feature class. Each feature instance is allocated its own group within the feature container group. This organization allows class-wide attributes to be attached to the class as a whole and instance-specific attributes to be attached to the appropriate feature instance.

NOTE: The decision to make a distinct group for each feature instance is based on the fact that there will be multiple datasets for a single instance in some circumstances (for example, index, TIN, etc), and placing all the datasets directly under the container group is likely to add confusion to the data organization from the human perspective at least (though suffixes might suffice to distinguish different instances for programming purposes).

The structure of the Feature Container group is shown in Table 10c-9 below. This Table also shows the feature instance group(s). The axis names are given in a dataset at the feature container level.

Metadata that is common to all instances of the feature class (such as dimensionality) is encoded at the feature container level and these metadata elements are listed in Table 10c-10. Metadata that is specific to feature instances (such as grid parameters) is encoded at the instance level and these elements are listed in Table 10c-12.

Product Specifications may add product-specific metadata attributes. The guidelines for additional metadata elements are the same as additional metadata elements in the root group (clause 10c-9.4).

Table 10c-9 – Structure of feature container groups

Group	HDF5 Category	Name	Data Type	Remarks / Data space
/(feature code)	attribute	See Table 10c-10	(see Table)	Single-valued attributes as described in Table 10c-10
	Dataset	axisNames	String	Array (1-D): 0..D-1 where D is the value of the *dimension* attribute Axes should be in major-minor order; that is, if storage is to be in row-major order the X/longitude axis should be first
	Dataset (optional)	coordinateSize	Integer	Array (1-D): 0..D-1 where D is the value of the *dimension* attribute The size of the coordinate encoding in bytes. Allowed values are 1, 2, 4, or 8. If this dataset is not present the coordinates must be encoded using 64 bits (8 bytes) for Float coordinates and 32 bits (4 bytes) for Integer coordinates
	Dataset (optional)	interpolationParameters	Float	Array (1-D) of interpolation parameters Required if and only if the value of attribute *interpolationType* is ‘biquadratic’ or ‘bicubic’
	Dataset (optional)	featureAttributeTable	Compound	Array (1-D) of index-based feature attributes (clause 10c-9.6.2). Required if and only if the value of attribute *dataCodingFormat* is 9
	Group	/(feature code).N		Container for each instance of a feature type. Numbered sequentially from 1 to *numInstances* (Table 10c-10). Zero-padding with leading zeros must be used so that the ‘N’ suffixes are all the same length. To accommodate expansion, an extra zero is recommended

NOTES:

1) “uncertainty” is the uncertainty in data values, position uncertainty (both horizontal and vertical) is encoded separately.
2) The length of the interpolationParameters dataset and sequence of parameters should be provided in the Product Specification.

Table 10c-10 – Attributes of feature container groups

No	Name	Camel Case	Mult	Data Type	Remarks and/or Units
	Data organization index	dataCodingFormat	1	Enumeration	Indication of the type of coverage in instances of this feature. Used to read the data (see Table 10c-4) 1: Time series at fixed stations 2: Regularly-gridded arrays 3: Ungeorectified gridded arrays 4: Moving platform 5: Irregular grid 6: Variable cell size 7: TIN 8: Time series at fixed stations (stationwise) 9: Feature oriented Regular Grid
	Dimension	dimension	1	Integer	The dimension of the feature instances This is the number of coordinate axes, not the rank of the HDF5 arrays storing coordinates or values. For example, a fixed stations dataset with positions in latitude and longitude will have dimension=2
	Common point rule	commonPointRule	1	Enumeration	The procedure used for evaluating the coverage at a position that falls on the boundary or in an area of overlap between geometric objects Values from CV_CommonPointRule (Part 8, Table 8-12)
	Horizontal position uncertainty	horizontalPositionUncertainty	1	Float	The uncertainty in horizontal coordinates. For example, -1.0 (unknown/inapplicable) or positive value (m)
	Vertical position uncertainty	verticalUncertainty	1	Float	The uncertainty in vertical coordinate(s). For example, -1.0 (unknown/inapplicable) or positive value (m)
	Time uncertainty	timeUncertainty	0..1	Float	Uncertainty in time values. For example, -1.0 (unknown/inapplicable) or positive value (s) Only for time series data
	Number of feature instances	numInstances	1	Integer	Number of instances of the feature (Records in the same time series or moving platform sequence are counted as a single instance, not as separate instances)
	(additional common attributes)				(As specified in Product Specification)
dataCodingFormat = 1					
	(none)				
dataCodingFormat = 2					
	Sequencing rule	sequencingRule.type	1	Enumeration	Method to be used to assign values from the sequence of values to the grid

		sequencingRule.scanDirection	1	String	coordinates Type and scan direction are encoded as separate attributes type: Enumeration CV_SequenceType (Part 8, Table 8-13) scanDirection: String <axisNames entry> (comma-separated). For example, “latitude, longitude”. Reverse scan direction along an axis is indicated by prefixing a ‘-‘ sign to the axis name
	Interpolation type	interpolationType	1	Enumeration	Interpolation method recommended for evaluation of the S100_GridCoverage Values: S100_CV_InterpolationMethod (Part 8, Table 8-14)
	Offset of data point in cell	dataOffsetCode	0..1	Enumeration	See clause 10c-9.6.1 1: XMin, YMin (“Lower left”) corner (“Cell origin”) 2: XMax, YMax (“Upper right”) corner 3: XMax, YMin (“Lower right”) corner 4: XMin, YMax (“Upper left”) corner 5: Barycenter (centroid) of cell
	Offset of data point in cell as vector	dataOffsetVector	0..1	Float	Array (1-D) 0..D-1 where D is the value of the dimension attribute Values must be real numbers in the range [0,1] See clause 10c-9.6.1
dataCodingFormat = 3					
	Interpolation type	interpolationType	1	Enumeration	Interpolation method recommended for evaluation of the S100_GridCoverage Values: S100_CV_InterpolationMethod (Part 8, Table 8-14)
dataCodingFormat = 4					
	(none)				
dataCodingFormat = 5					
	Sequencing rule	sequencingRule.type	1	Enumeration	Method to be used to assign values from the sequence of values to the grid coordinates Type and scan direction are encoded as separate attributes type: Enumeration CV_SequenceType (Part 8, Table 8-13) scanDirection: String <axisNames entry> (comma-separated). For example, “latitude, longitude”. Reverse scan direction along an axis is indicated by prefixing a ‘-‘ sign to the axis name
		sequencingRule.scanDirection	1	String	
	Interpolation type	interpolationType	1	Enumeration	Interpolation method recommended for evaluation of the S100_GridCoverage Values: S100_CV_InterpolationMethod (Part 8, Table 8-14)

	Offset of data point in cell	dataOffsetCode	0..1	Enumeration	See clause 10c-9.6.1 1: XMin, YMin (“Lower left”) corner (“Cell origin”) 2: XMax, YMax (“Upper right”) corner 3: XMax, YMin (“Lower right”) corner 4: XMin, YMax (“Upper left”) corner 5: Barycenter (centroid) of cell
	Offset of data point in cell as vector	dataOffsetVector	0..1	Float	Array (1-D) 0..D-1 where D is the value of the dimension attribute Values must be real numbers in the range [0,1] See clause 10c-9.6.1
dataCodingFormat = 6					
	Sequencing rule	sequencingRule.type	1	Enumeration	Method to be used to assign values from the sequence of values to the grid coordinates Type and scan direction are encoded as separate attributes type: Enumeration CV_SequenceType (Part 8, Table 8-13) scanDirection: String <axisNames entry> (comma-separated). For example, “latitude, longitude”. Reverse scan direction along an axis is indicated by prefixing a ‘-‘ sign to the axis name
		sequencingRule.scanDirection	1	String	
	Interpolation type	interpolationType	1	Enumeration	Interpolation method recommended for evaluation of the S100_GridCoverage Values: S100_CV_InterpolationMethod (Part 8, Table 8-14)
	Offset of data point in cell	dataOffsetCode	0..1	Enumeration	See clause 10c-9.6.1 1: XMin, YMin (“Lower left”) corner (“Cell origin”) 2: XMax, YMax (“Upper right”) corner 3: XMax, YMin (“Lower right”) corner 4: XMin, YMax (“Upper left”) corner 5: Barycenter (centroid) of cell
	Offset of data point in cell as vector	dataOffsetVector	0..1	Float	Array (1-D) 0..D-1 where D is the value of the dimension attribute Values must be real numbers in the range [0,1] See clause 10c-9.6.1
dataCodingFormat = 7					
	Interpolation type	interpolationType	1	Enumeration	Interpolation method recommended for evaluation of the S100_GridCoverage Values: S100_CV_InterpolationMethod (Part 8, Table 8-14)
dataCodingFormat = 8					
	(none)				
dataCodingFormat = 9					

	Offset of data point in cell	dataOffsetCode	0..1	Enumeration	See clause 10c-9.6.1 1: XMin, YMin (“Lower left”) corner (“Cell origin”) 2: XMax, YMax (“Upper right”) corner 3: XMax, YMin (“Lower right”) corner 4: XMin, YMax (“Upper left”) corner 5: Barycenter (centroid) of cell
	Offset of data point in cell as vector	dataOffsetVector	0..1	Float	Array (1-D) 0..D-1 where D is the value of the dimension attribute Values must be real numbers in the range [0,1] See clause 10c-9.6.1
	Sequencing rule	sequencingRule.type sequencingRule.scanDirection	1 1	Enumeration String	Method to be used to assign values from the sequence of values to the grid coordinates Type and scan direction are encoded as separate attributes type: Enumeration CV_SequenceType (Part 8, Table 8-13) scanDirection: String <axisNames entry> (comma-separated). For example, “latitude, longitude”. Reverse scan direction along an axis is indicated by prefixing a ‘-‘ sign to the axis name
(any dataCodingFormat value)					
	(additional attributes)				(As specified in Product Specification)

10c-9.6.1 Location of data point within cell

Product Specifications may require their data products to indicate the relative location of the data point corresponding to a grid cell in relation to the corners of the cell. The location can be indicated using either the *dataOffsetCode* or *dataOffsetVector* attribute. These attributes can be used only with grid-based coverages and not with time series, TIN, or moving platform data. Product Specifications may use either *dataOffsetCode* or *dataOffsetVector* but not both.

Product Specifications in which the data point is located at the (XMin, YMin) grid point need not use either *dataOffsetCode* or *dataOffsetVector*.

The attribute *dataOffsetCode* can be used only with two-dimensional grids. It indicates whether the data point is one of the four cell corners or the centre of the cell. Note that the definitions of the codes indicting the corners are in terms of X and Y grid coordinates relative to the grid origin. (This means that in a grid with its X axis directed from east to west and Y axis from north to south the “lower left” corner is different from the “lower left” corner in a grid with X axis directed west to east and Y axis south to north.)

The attribute *dataOffsetVector* is intended for use with higher-dimension grids or in cases where the data point location is not at one of the corners or the centre of the cell. The values in this array indicate the relative offset along each axis of the data point from the grid point whose grid coordinates are closest to those of the grid origin. In a two-dimensional grid, this will be the point with smallest X and Y grid coordinates. Again, it should be noted that the direction of the axes and the location of the grid origin determines which corner is the cell origin. Each offset is relative to the dimension of the cell along the corresponding axis. The order of values in *dataOffsetVector* must correspond to the order of axes in the *axisNames* array (Table 10c-9).

10c-9.6.2 Feature attribute table

This element is an optional record with the name "featureAttributeTable". The element is only used in the feature container group if the dataCodingFormat = 9 (Feature oriented Regular Grid). In all other cases, the element must not be used. The "featureAttributeTable" element can only ever be used in conjunction with the “values”-record of the "data values group(s)".

The "featureAttributeTable" element corresponds to the HDF5 data type of a CompoundDataset as a 1D array. It always consists of a fixed column with the name "id" and any number of additional columns.

Column id

The "id" column always contains one-to-one values of the unsigned integer data type. The values are always greater than zero (>0), since the value zero (0) is specified as a nodata value. The values do not have to correspond to any particular order. They can be chosen randomly. This allows using the identification value of an external data source; for example, a database. The column "id" always contains a value from the "values-record" of the "data values group(s)" of the "feature instance group(s)".The value serves as a link between the raster geometry of a feature and its descriptive data in the "featureAttributeTable". The one-to-one values of the "id" column may occur redundantly in the pixels in the raster image, but must always describe exactly one feature geometry. The effect of the redundancy in the raster image can be significantly reduced by compression within the HFD5 dataset.

Additional columns

The number of additional columns depends on the information to be transported. The number must always be greater than zero (>0). Table 10c-8 defines that the name of the "feature container group" must always correspond to the camelCase name of an object of the type "Feature Type" of the IHO Geospatial Information (GI) Registry, Data Dictionary Register. The names of the other columns are based on the attributes of the selected feature type of the Data Dictionary Register. Due to the mapping form of the attributes in the HDF5 format, only simple data types (see clause 10c-7) can be used. For the mapping of complex data types, a corresponding notation by means of a dot is necessary. The respective levels of the complex data type are separated by a dot up to the simple data type; for example, "surveyDateRange.dateEnd" & "surveyDateRange.dateStart".

10c-9.7 Feature instance group

The feature instance groups are contained within the feature container groups. The structure of a feature instance group is defined in Table 10c-11. The attributes that are specific to each feature instance are defined in the Table following (Table 10c-12) and consist of information that may vary for different instances in the same dataset, such as extent, location, time, and grid size.

Table 10c-11 – Structure of feature instance groups

Group	HDF5 Category	Name	Data Type	Remarks / Data space
/(feature code).N For example: SurfaceCurrent.01	attributes	See Table 10c-12	(see Table)	Single-valued attributes as described in Table 10c-12
	Dataset (optional)	domainExtent.polygon	Compound (Float, Float)	Spatial extent of the domain of the coverage Array (1-d): i=0, P Components: <longitude, latitude> or <X, Y> (coordinates of bounding polygon vertices as a closed ring; that is, the first and last elements will contain the same values) Either this or the bounding box attribute must be populated. For irregular arrays, this dataset must specify the polygon indicating the area for which data are provided
	Dataset (optional)	domainExtent.verticalElement	Compound (Integer X 2, Float X 2)	Array (1-d) of compound elements each providing a grid location and maximum, minimum vertical extents at the location The components of the compound type are: gridX, gridY: Integer (grid point numbers along X/longitude and Y/latitude axes) minimumValue, maximumValue (Float): minimum and maximum Z values at the grid point specified by gridX and gridY Applicable only to 3-D grids. Either this dataset or the verticalExtent attribute (Table 10c-12) must be populated for 3-D grids
	Dataset (optional)	extent	Compound (Integer X D)	1-D array, of compound elements, 2 rows. Row 0 gives the "low" values, row 1 the "high" values The area of the grid for which data are provided. (Part 8 Figure 8-24) Components of compound type are named according to the axis names in the axisNames dataset
	Dataset (optional)	uncertainty	Compound (String, Float)	Array (1-d): i = 0, (up to) NA_F Code and uncertainty of data values For example, ("surfaceCurrentSpeed", 0.1) The number of attributes for this feature class (NA_F) may be determined from Group_F

	Dataset (optional)	cellGeometry	Compound (String, Float X 2, Integer X 1)	Cell geometry. Array (1-d) of length the same as the *axisNames* array defined above (this means that if present, this dataset encodes all the axes including latitude, longitude, etc) Conditional, required only for regular grids (dataCodingFormat = 2 or 9) using coordinate reference systems with axes other than (latitude, longitude, vertical), or with more than 3 dimensions This array serves to extend the information encoded in the grid parameter attributes (origin, spacing, number of points) defined in Table 10c-12 (Attributes of feature instance group) for data products which use higher-dimensional grids or non-standard coordinate axes Components: axisName: string (an entry in the *axisNames* array defined above) gridOrigin: Float (the origin of the axis named in the axisName component) gridSpacing: Float (Cell spacing for the named axis) numPoints: Integer (the number of grid lines along the named axis)
	Group (optional)	/Group_TL		Tile information. Conditional, required if the Product Specification specifies tiling.
	Group (optional)	/Group_IDX		Spatial indexing method. Conditional, required if the Product Specification specifies spatial indexing.
	Group (optional)	/Positioning		Positioning information. Coordinates of data values. Conditional, required if dataCodingFormat is not 2 (Regular grid)
	Group	/Group_nnn		Data Values group(s).

Table 10c-12 – Attributes of feature instance groups

No	Name	Camel Case	Mult	Data Type	Remarks and/or Units
	Bounding box	westBoundLongitude	0..1	Float	The geographic extent of the grid, as a bounding box Ref. domainExtent: EX_GeographicExtent > EX_GeographicBoundingBox Either this or the domainExtent dataset must be populated The bounds must either all be populated or all omitted The minimum/maximum latitude and longitude of the data is based on a Geographic CRS that uses the same horizontal datum as the CRS defined for the data set which might be projected. The unit must be degrees.
		eastBoundLongitude	0..1	Float	
		southBoundLatitude	0..1	Float	
		northBoundLatitude	0..1	Float	

	Number of time records	numberOfTimes	0..1	Integer	The total number of time records Time series data only. For dataCodingFormat = 8, this variable may be overridden by the corresponding one in the values group attributes (Table 10c-19)
	Time interval	timeRecordInterval	0..1	Integer	The interval between time records. Units: Seconds Time series data only. For dataCodingFormat = 8, this variable may be overridden by the corresponding one in the values group attributes (Table 10c-19)
	Valid Time of Earliest Value	dateTimeOfFirstRecord	0..1	Character	The validity time of the earliest time record. Units: DateTime Time series data only
	Valid Time of Latest Value	dateTimeOfLastRecord	0..1	Character	The validity time of the latest time record. Units: DateTime Time series data only
	Vertical extent	verticalExtent.minimumZ	0..1	Float	Vertical extent of 3-D grids minimumZ, maximumZ: Minimum and maximum values of the grid's spatial extent along the vertical direction. They are encoded as separate attributes
		verticalExtent.maximumZ	0..1	Float	
	Number of groups	numGRP	1	Integer	The number of data values groups contained in this instance group.
	(additional attributes specific to data product)	(as defined in Product Specification)			
dataCodingFormat = 1					
	Number of fixed stations	numberOfStations	1	Integer	The number of fixed stations
dataCodingFormat = 2 or 9					
	Longitude of grid origin	gridOriginLongitude	1	Float	The longitude of the grid origin. Unit: Arc Degrees
	Latitude of grid origin	gridOriginLatitude	1	Float	The longitude of the grid origin. Arc Degrees
	Vertical grid origin	gridOriginVertical	0..1	Float	The grid origin in the vertical dimension. Only for 3-D grids. Units specified by Product Specifications
	Grid spacing, long.	gridSpacingLongitudinal	1	Float	Cell size in the X/longitude dimension. This is the X/longitudinal component of the offset vector (S-100 Part 8, clause 8-7.5). Units: Arc Degrees
	Grid spacing, lat.	gridSpacingLatitudinal	1	Float	Cell size in the Y/latitude dimension. This is the Y/latitudinal component of the offset vector (S-100 Part 8, clause 8-7.5). Units: Arc Degrees
	Grid spacing, Z	gridSpacingVertical	0..1	Float	Cell size in the vertical dimension. Only for 3-D grids. Units specified by Product Specifications
	Number of points, long.	numPointsLongitudinal	1	Integer	Number of grid points in the X/longitude dimension. (iMax)
	Number of points, lat.	numPointsLatitudinal	1	Integer	Number of grid points in the Y/latitude dimension. (jMax)
	Number of points, vertical	numPointsVertical	0..1	Integer	Number of grid points in the vertical dimension. (kMax)

	Start sequence	startSequence	1	String	Grid coordinates of the grid point to which the first in the sequence of values is to be assigned. The choice of a valid point for the start sequence is determined by the sequencing rule. Format: n, n… (comma-separated list of grid points, one per dimension – For example, 0,0)
dataCodingFormat = 3					
	Nodes in grid	numberOfNodes	1	Integer	The total number of grid points
dataCodingFormat = 4					
	Number of stations	numberOfStations	1	Integer	Value is always 1
dataCodingFormat = 5 or 6					
	Longitude of grid origin	gridOriginLongitude	1	Float	The longitude of the grid origin. Unit: Arc Degrees
	Latitude of grid origin	gridOriginLatitude	1	Float	The longitude of the grid origin. Arc Degrees
	Vertical grid origin	gridOriginVertical	0..1	Float	The grid origin in the vertical dimension. Only for 3-D grids. Units specified by Product Specifications
	Grid spacing, long.	gridSpacingLongitudinal	1	Float	Cell size in the X/longitude dimension. This is the X/longitudinal component of the offset vector (S-100 Part 8, clause 8-7.5). Units: Arc Degrees For variable cell size grids this is the unit cell size (the size of the smallest cell in this dimension)
	Grid spacing, lat.	gridSpacingLatitudinal	1	Float	Cell size in the Y/latitude dimension. This is the Y/latitudinal component of the offset vector (S-100 Part 8, clause 8-7.5). Units: Arc Degrees For variable cell size grids this is the unit cell size
	Grid spacing, Z	gridSpacingVertical	0..1	Float	Cell size in the vertical dimension. Only for 3-D grids. Units specified by Product Specifications. For variable cell size grids this is the unit cell size
	Nodes in grid	numberOfNodes	1	Integer	The total number of grid points
	Start sequence	startSequence	1	String	Grid coordinates of the grid point to which the first in the sequence of values is to be assigned. The choice of a valid point for the start sequence is determined by the sequencing rule. Format: n, n… (comma-separated list of grid points, one per dimension – for example, 0,0)
dataCodingFormat = 7					
	Nodes in grid	numberOfNodes	1	Integer	The total number of grid points
	Triangles in grid	numberOfTriangles	1	Integer	The total number of triangles in the TIN
dataCodingFormat = 8					
	Number of fixed stations	numberOfStations	1	Integer	The number of fixed stations
(any dataCodingFormat value)					

	(additional attributes)				(As specified in Product Specification)

NOTES:

1) The type-specific attributes for regular and variable cell size grids are the same except that the parameters giving the number of points in each dimension are replaced by the total number of nodes in the grid.
2) Attributes "Valid time of earliest value" and "Valid time of latest value" provide the *temporalElement* component of the domainExtent attribute in the grid model (S-100 Part 8, Figures 8-22, 8-23 and 8-24).
3) Product Specifications may require use of one or the other of the domainExtent or boundingBox attributes, depending on whether spatial extents of feature instances are definitely known to be rectangular in the coordinate system or definitely known to be of irregular shape.

Page intentionally left blank

10c-9.7.1 Overriding attributes

A feature instance group may also carry any of the following attributes defined in higher-level groups. The attribute value assigned in the feature instance group overrides the value in the higher group.

- The "Vertical datum reference" (verticalDatum) attribute from the Root group;
- Any attribute from the Feature Container group, **except** "Number of feature instances" (numInstances).

Product Specifications may prohibit attribute overriding if not required for their products.

NOTES:

1) Attribute overriding is intended to allow certain products to encode variations of feature types in the same data file, for example, if an application schema defines a feature which can have either regular grid or fixed station information, and therefore may need different metadata attributes. Product Specification authors should note however that this issue can be resolved in application schemas by defining appropriate specializations of the feature class, which would be distinct feature types, and therefore encoded in different feature containers.
2) Attribute overriding also allows production-time differences, such as different vertical datums for different instances. While this is possible, its practice should be avoided in order to reduce the possibility of human error in application development as well as by the end-user.

10c-9.7.2 Example of container and instance structure

The figure below depicts the structure of a hypothetical data file containing 3 instances of the **SurfaceCurrent** feature type.

- The vertical panel on the left shows the overall structure. The data product consists of 2 features (**SurfaceCurrent** and **WaterLevel**). Each is represented by a group just under the root group. The Feature Information group described earlier (clause 10c-9.5) is also shown.
- The Feature Container group named **SurfaceCurrent** contains 3 instances of the **SurfaceCurrent** feature type (hypothetically, data for 3 separate places, each with a local coverage grid). Each instance contains subgroups (Group_001, etc) for time series data.
- Locations are encoded in the **geometryValues** dataset in the **Positioning** group (panel at top right). The **axisNames** panel to its left names the components of the **geometryValues** (that is., the coordinate axes).
- The **SurfaceCurrent** panel in the the middle shows the metadata attributes common to all instances, which are attached to the **SurfaceCurrent** feature container group.
- The two panels at the bottom show the instance-specific metadata for the feature instances **SurfaceCurrent.01** and **SurfaceCurrent.02**.

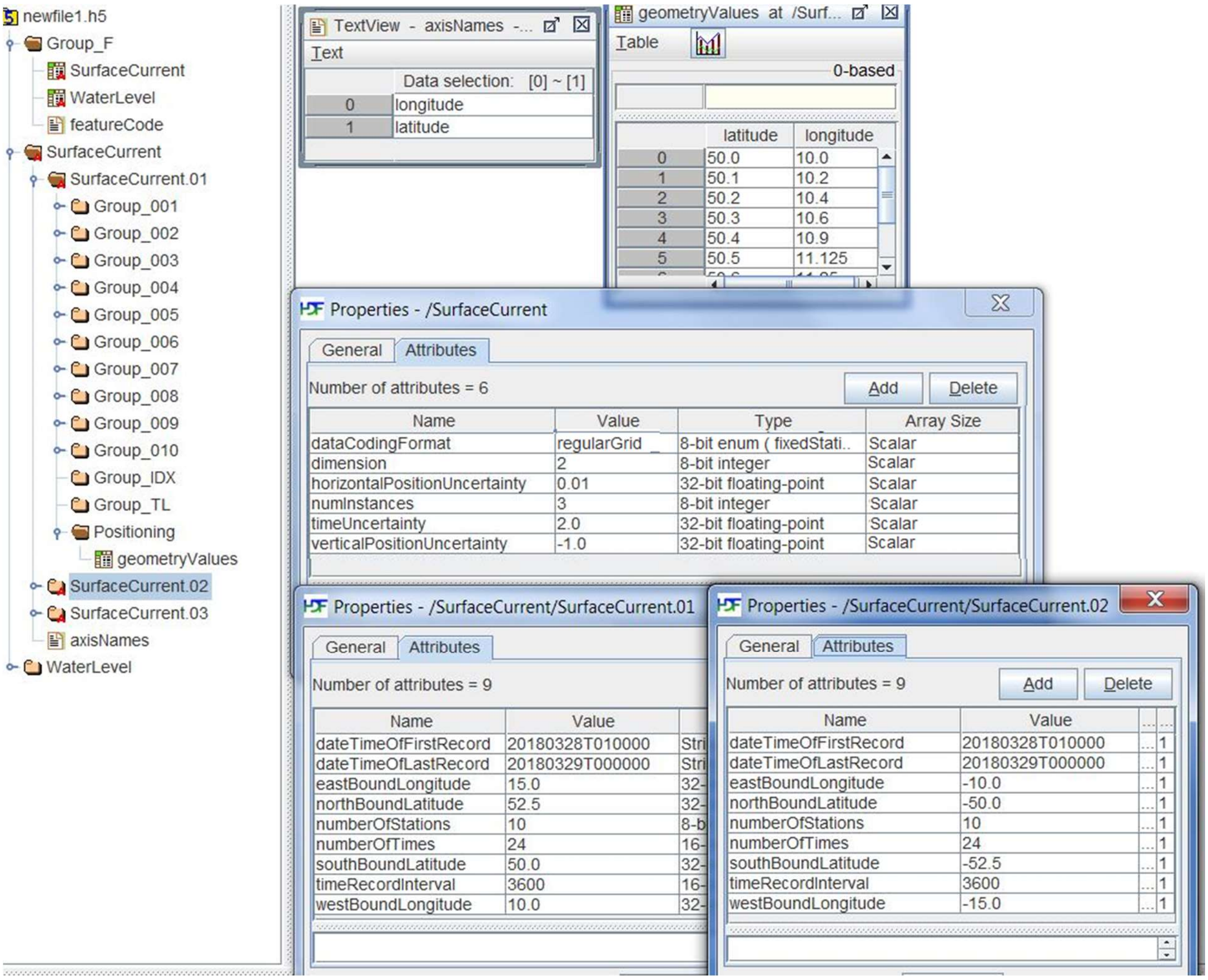


Figure 10c-9 – Illustrative example of dataset structure.

10c-9.8 Tiling information group

This group encodes information about the tiling scheme used in the (S-100) dataset. It is present if and only if the data is encoded in more than a single tile. Some tiling schemes are described in S-100 Part 8, clause 8-6. This edition of the HDF5 profile supports only two tilings: simple grid and variable density simple grid. In both cases, the extents of the tiles are specified in terms of their bounding boxes (Table 10c-12).

The spatial union of tile surfaces must cover all the features in the (S-100) dataset, but the converse is not a requirement. (Informally, this means that there may be parts of tiles that are not covered by the geometry of any feature in the dataset, but not vice versa – there cannot be parts of feature geometry that are not covered by at least one tile.)

Note that tiling is not quite the same concept as “chunking”, as the latter is defined in HDF5 and NetCDF – tiles are coordinate-based geographical partitions, while chunking defines slices of HDF5 datasets for storage and retrieval performance optimization.

Table 10c-13 – Tiling information group

Group	HDF5 Category	Name	Data Type or HDF Category	Remarks / Data space
/Group_TL	Attribute	numTiles	Integer	Number of tiles value > 0

	Attribute	tilingScheme	Enumeration	1: Simple grid 2: Variable-density simple grid (Product Specification must pick one)
	Dataset	tiles	Array Compound (Float X 4, Integer)	Bounding boxes of tiles. Components: westBoundLongitude: Float eastBoundLongitude: Float southBoundLatitude: Float northBoundLatitude: Float tileID: Integer (tile identifier)

The details of tiling methods are left to Product Specifications in this edition of S-100. This profile does not specify an ordering for the tiles, nor does it control the use or non-use of hierarchical tiling schemes. S-100 Part 8, clause 8-6 requires that any tiling scheme used must be completely described as part of the Product Specification for a particular data product. This includes the dimensions, location and data density of tiles as well as a tile identification mechanism (tileID).

10c-9.9 Indexes group

The indexes group encodes spatial indexing information, if used by the Product Specification. This group is encoded if and only if the Product Specification prescribes a spatial indexing method and requires explicit encoding of the spatial index.

Table 10c-14 – Indexes group

Group	HDF5 Category	Name	Data Type or HDF Category	Remarks / Data space
/Group_IDX	Attribute	indexingMethod	Enumeration	Spatial indexing method. (Described in Product Specifications)
	Dataset(s)	spatialIndex	(Depends on indexing method)	Data encoding the spatial index. (Described in Product Specifications)

The details of indexing methods and the structure of index datasets are left to Product Specifications in this edition of S-100.

10c-9.10 Positioning group

Depending of the data coding format, there can be a positioning group, Positioning. This group contains no attributes, it contains a coordinates dataset, which is an array of compound type with components named the same as the *axisNames* dataset in the Feature Container group. This group is used for values of *dataCodingFormat* of 1, 3, 4, 7 and 8 (clause 10c-9.3). It is not used for *dataCodingFormat* = 2 (regular grids), 5 (irregular grid), 6 (variable cell size grid), or 9 (Feature oriented Regular Grid).

The traversal order for grids of different types is specified by the carrier metadata attribute *sequencingRule* in the feature container group. Traversal order is not used for fixed station, moving platform, or TIN, or fixed station (stationwise) data (*dataCodingFormat* = 1, 4, 7 or 8).

The dimensionality D of the data is given by the *dimension* metadata attribute in the feature container group.

10c-9.10.1 Spatial representation strategy

For regularly gridded data (dataCodingFormat = 2 or 9), the number of grid points in each dimension, grid spacing, and grid origin are encoded in metadata attributes. (For example, for 2-D grids, the metadata attributes *numPointsLongitudinal* and *numPointsLatitudinal* encode the points along the longitude and latitude axes.) Given these parameters and the indexes of a point in the grid, the position of the point can be computed by simple formulae.

For fixed station time series data, ungeorectified gridded data, moving platform data, triangulated irregular networks and fixed station (stationwise) time series data (that is, when dataCodingFormat is 1, 3, 4, 7 or 8), the location of each point must be specified individually. This is accomplished in an HDF5 dataset in the “Positioning” group, which gives the individual location coordinates (for example, longitude and latitude) for each location. For fixed station time series and fixed station (stationwise) time series data, the longitude and latitude values are the positions of the stations; the number of stations is *numberOfStations*. For ungeorectified gridded data, the values are the positions of each point in the grid; the number of grid points is *numberOfNodes*. For moving platform data, values are the positions of the platform at each time; the number of platforms is *numberOfStations*.

For irregular grid and variable cell size coverages (dataCodingFormat 5 and 6), the storage format uses the same metadata as for regular grids plus HDF5 datasets indicating which cells are populated or aggregated respectively. The latter datasets encode the locations of cells in terms of grid point or cell address in grid coordinates – that is, the indexes in the grid, or the Morton code – not the geographic (latitude/longitude) coordinates. The sequencing and axis order needed for interpretation of the grid coordinates as geographic coordinates are given by the *sequencingRule* and *scanDirection* attributes respectively. By combining this information with the grid parameters provided in metadata, the position of populated cells/points can be computed with slightly more complex formulae than for regularly gridded data.

The Table below summarizes the strategies for storage of coordinate information.

Table 10c-15 – Positioning dataset types and dimensions for different coverage types

Type of coverage	dataCoding Format	Structure of coordinates dataset
Fixed Stations	1	1-dimensional Array, length = numberOfStations
Regular Grid	2	not used
Ungeorectified Grid	3	1-dimensional Array, length = numberOfNodes
Moving Platform	4	1-dimensional Array, length = numberOfTimes
Irregular Grid	5	not used
Variable cell size	6	not used
TIN	7	1-dimensional Array, length = numberOfNodes
Fixed Stations (Stationwise)	8	1-dimensional Array, length - numberOfStations
Feature oriented Regular Grid	9	not used

NOTE: Multiple moving platforms can be encoded as different feature instances.

10c-9.10.2 Data structures for storing position information for grid points

The number of positions is computed as specified in Table 10c-4 in clause 10c-9.3.

Table 10c-16 – Positioning group

Group	HDF5 Category	Name	Data Type	Data Space
/Positioning	Dataset	geometryValues	Compound (Float X D)	Array (1-dimensional) of size dependent on dataCodingFormat, see Table 10c-15 Components of compound type are named according to the axis names (for example, ‘latitude’, ‘longitude’, ‘Z’, etc) The dimension D and the component names are specified in the feature container group *dimension* attribute and *axisNames* dataset respectively (Tables 10c-10 and 10c-9)
	Dataset	triangles (optional)	Array (Integer)	Array (2-d): dimensions numberOfTriangles X 3 Each row encodes a triangle as the indexes of 3 coordinates in the *geometryValues* dataset Required only for dataCodingFormat = 7 (TIN)
	Dataset	adjacency (optional)	Array (Integer)	Array (2-d): dimensions numberOfTriangles X 3 Each row encodes the triangles adjacent to any given triangle by specifying their indexes in the triangles dataset adjacency[i][0] = triangle adjacent to the edge specified by triangles[i][0] & triangles[i][1] adjacency[i][1] = triangle adjacent to edge triangles[i][1] & triangles[i][2] adjacency[i][2] = triangle adjacent to edge triangles[i][2] & triangles[i][0] Elements for edges without adjacent triangles are filled with the value -1 Applicable only for dataCodingFormat = 7 (TIN), but optional even for TIN.

10c-9.11 Data values groups

The structure of data values content is analogous to that of positioning content, except that regular grid data values (dataCodingFormat = 2 or 9) are stored as a D-dimensional array corresponding to the axis order in the *axisNames* dataset in the Feature Container group (major index precedes minor index). The dimensionality D is encoded in the *dimension* attribute of the Feature Container group.

EXAMPLE: For two-dimensional regularly gridded data, the value arrays are two dimensional, with dimensions numPointsLongitudinal and numPointsLatitudinal.

For fixed station time series data, ungeorectified gridded data, moving platform data, triangulated irregular networks and fixed station (stationwise) time series data (that is, when dataCodingFormat is 1, 3, 4, 7 or 8), the data values are stored as 1-dimensional datasets of length given by the numberOfTimes, numberOfNodes or numberOfStations metadata attribute of the feature instance group (Table 10c-12) depending on the dataCodingFormat.

For irregular grid coverages (dataCodingFormat=5), the storage of data values is the same as for ungeorectified grids etc (that is, a 1-dimensional array of value records, length = numberOfNodes) but the value group includes a dataset that specifies the grid point or cell address associated to each entry in the values array. This second dataset uses grid coordinates – that is, the indexes in the grid, or the Morton code – not the geographic (latitude/longitude) coordinates. The sequencing and axis order needed for interpretation of the grid coordinates as geographic coordinates are given by the *sequencingRule* and *scanDirection* attributes respectively.

For variable cell size coverages (dataCodingFormat=6) the storage of data values is the same as for irregular grid coverages but the values groups contains the grid index dataset used by irregular grids as well as a dataset indicating which cells are aggregated into larger cells.

The various datasets and their components are described in the following Table.

Table 10c-17 – Values dataset type and size for different data encoding formats

Type of coverage	dataCoding Format	Structure of values and auxiliary HDF5 datasets	HDF5 Dataset components
Fixed Stations	1	values: 1-dimensional Array, length = numberOfStations	Compound, one component for each attribute specified in the corresponding feature information dataset in the Feature Information group (Table 10c-8) Component name: attribute code as specified in the feature information dataset Component type: Any appropriate HDF5 datatype consistent with the attribute datatype specified in the Feature Information dataset
Regular Grid	2	values: D-dimensional array, dimensions specified by: 2-D: numPointsLatitudinal X numPointsLongitudinal 3-D: numPointsLatitudinal X numPointsLongitudinal X numPointsVertical If *cellGeometry* is present in feature instance group: product of all cellGeometry[i].numPoints values.	As for fixed stations
Ungeorectified Grid	3	values: 1-dimensional Array, length = numberOfNodes	As for fixed stations
Moving Platform	4	values: 1-dimensional Array, length = numberOfTimes	As for fixed stations
Irregular Grid	5	values: 1-dimensional Array, length = numberOfNodes	As for fixed stations. Ordered according to the sequence rule specified by the *sequencingRule* and *scanDirection* attributes of the Feature Container group (Table 10c-10)
		gridIndex: 1-dimensional Array, length = numberOfNodes (dataset attribute codeSize: Integer - gives the length of the bitfield)	Element type: bitfield (length determined by grid dimensions) Order of element corresponds to the values array Each element contains the code of the cell (grid point) according to the sequence rule specified by the *sequencingRule* and *scanDirection* attributes. For example, the Morton code of the cell
Variable cell size	6	values: 1-dimensional Array, length = numberOfNodes	As for fixed stations

		gridIndex: 1-dimensional Array, length = numberOfNodes (dataset attribute codeSize: Integer - gives the length of the bitfield)	(As for the *gridIndex* Array for irregular grids) For cells that aggregate multiple unit cells, use the first cell (grid point) encountered in the sequencing order. For example, the Morton code of the cell
		cellScale: 1-dimensional Array, length = numberOfNodes	Element type: Compound Order of elements corresponds to the values array Components of the compound type are named according to the axis names in the axisNames dataset in the Feature Container group Each component is of type Integer and gives the number of cells aggregated along the named axis
TIN	7	values: 1-dimensional Array, length = numberOfNodes	As for fixed stations
Fixed Station (Stationwise)	8	values: 1-dimensional Array, length = numberOfTimes	As for fixed stations
Feature oriented Regular Grid	9	values: D-dimensional array, dimensions specified by: 2-D: numPointsLatitudinal X numPointsLongitudinal If *cellGeometry* is present in feature instance group: product of all cellGeometry[i].numPoints values	As for Regular Grid The name of the feature container group must correspond to a camelCase name of a feature type object from the IHO Geospatial Information (GI) Registry, Data Dictionary Register

NOTES:

2) 64-bit unsigned integers for gridIndex arrays allow 4-D grids with a maximum of 2^{16} - 1 (65,535) points/cells in each dimension.

3) The *gridIndex* datasets have an integer attribute named *codeSize* that gives the length (in bits) of the bitfield that contains the index. This depends on the type of code and the number of dimensions. For example, a 2-D grid with 8 points in each dimension needs 6-bit Morton codes.

4) The size of the bitfield is calculated by multiplying the number of bits needed to accommodate the largest dimension by the number of dimensions (D). To reduce complexity each dimension is allocated the same number of bits in the bitfield. For example, a 200 X 1000 array is given a 20-bit bitfield, calculated as:

$$\text{codesize} = 2 \times \max(\lceil \log_2 200 \rceil, \lceil \log_2 1000 \rceil).$$

The Figure that follows depicts *gridIndex* and *cellScale* arrays for an irregular grid (left) and variable cell size array (right). Both use Morton codes and 2-D grids of (nominally) 4×4 cells in each dimension. Note that in the Figure it is the cells rather than grid points that are assigned codes. The panels on the left describe an irregular grid with 11 populated cells. The panels on the right describe a variable cell size grid with two aggregate cells, each aggregating 2×2 unit cells.

The grids themselves are depicted below the panels, with the Morton codes shown in the respective cells[3]. The example on the right also indicates the scaling of each cell in parentheses (it is assumed that the scaling is the same in all dimensions; that is, cells 0100 and 1000 each aggregate 2×2 regions of the grid).

For the irregular grid example, the missing cells are not shown in the grid. For the variable cell size example, the greyed cells are aggregated with cells 0100 or 1000.

For variable cell size grids, this profile specifies the size of aggregated cells in terms of the number of unit cells they cover in each direction, instead of applying the same zoom factor in each dimension as depicted in the example at the bottom right of the Figure. This is for the better accommodation of rectangular and odd-shaped aggregations. Odd-shaped regions must be split into multiple rectangular aggregations. (Using rectangular aggregations has an associated extra storage cost.)

Further optimizations may be addressed in future editions of this profile.

[3] The two grid depictions at the bottom of the Figure are from “Elevation Surface Model Standardized Profile” (DGIWG 116-1) Ed. 1.0.1, Defence Geospatial Information Working Group (10 June 2014).

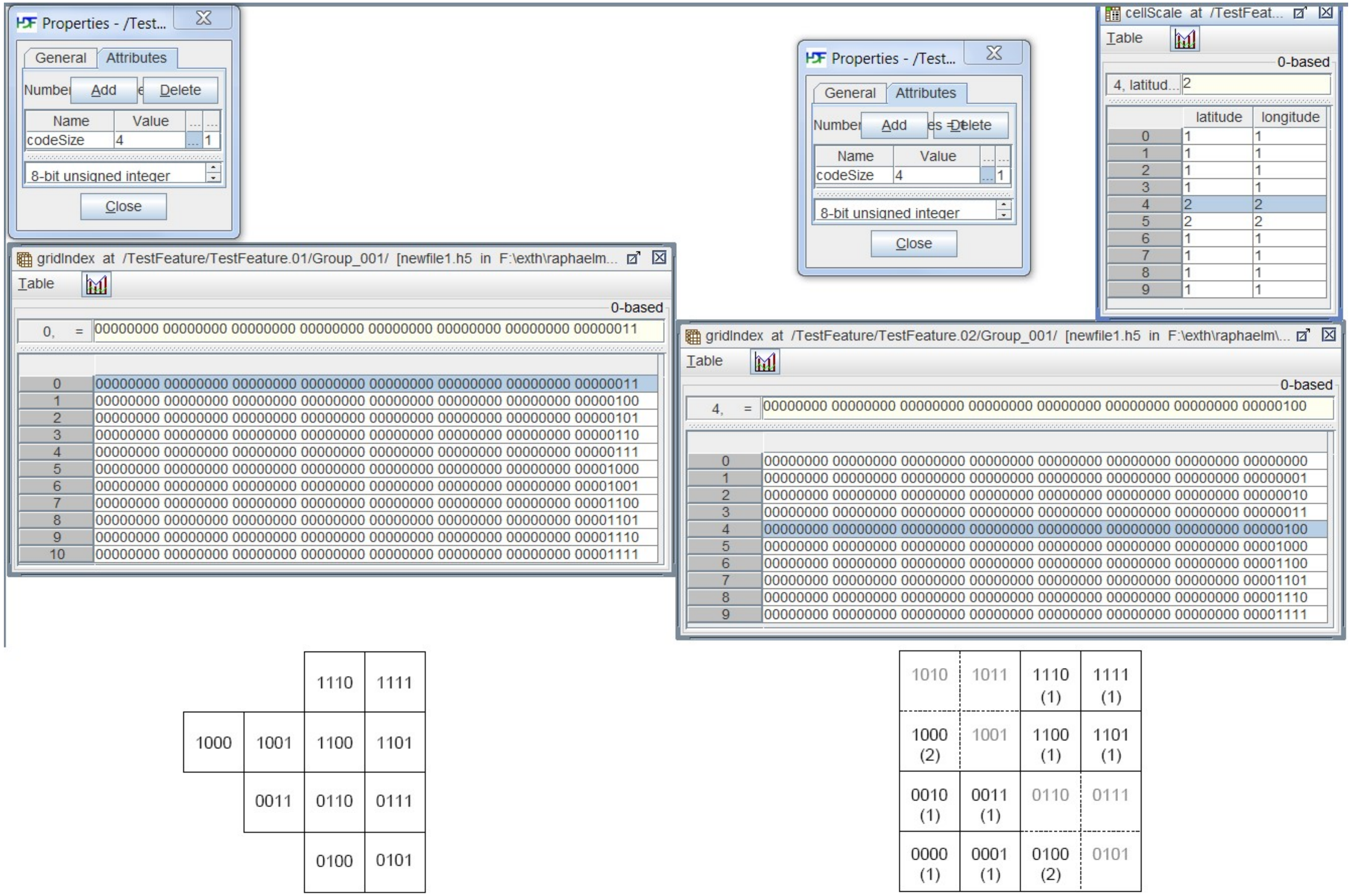


Figure 10c-10 – Illustrative examples of grid index array for irregular grids (left) and grid index and cell scale arrays for variable cell size grids (right)

The structure of the data values groups can now be described. Each group is structured as depicted in the Table below.

Table 10c-18 – Structure of values groups

Group	HDF5 Category	Name	Data Type	Data Space
/Group_NNN	Attribute	See Table 10c-19	(see Table)	Single-valued attributes as described in Table 10c-19
	Dataset	values	Compound	Array of Compound type, with array rank depending on dataCodingFormat and spatial dimension, as described in Table 10c-17
	Dataset	gridIndex	Bitfield	Required for dataCodingFormat = 5 or 6 Described in Table 10c-17
	Dataset	cellScale	Compound	Required for dataCodingFormat = 6 Described in Table 10c-17

Time series data for all except the moving platforms and fixed station (stationwise) format (dataCodingFormat = 4, 8) are encoded in successive groups contained within the instance group.

The sub-Groups each contain a date-time value, and the value record arrays. For dataCodingFormat = 2, 3, 5, or 6, the date-time is for the entire grid. The data value arrays are two dimensional, with a number of columns (numCOLS) and rows (numROWS). For a time series, the data values will be for each time in the series. For a grid, the speed and direction values will be for each point in the grid.

The Groups are numbered 001, 002, etc, up to the maximum number of Groups, numGRP. For all coverage types except moving platforms and fixed station (stationwise) time series data, the number of Groups is the number of time records. For moving platform data, there is only one Group, corresponding to a single platform; additional platforms can be accommodated in additional feature instances. For fixed stations (stationwise) data, the number of Groups is the number of stations.

The number of individual Groups is given by the metadata variable, *numGRP*. The uniform time interval between individual times is given by the metadata variable *timeRecordInterval*.

Values which represent different times are stored sequentially, from oldest to newest. The initial date-time value is contained in a metadata attribute (Table 10c-12). By knowing the time interval between each record, the time applicable to each value can be computed.

Groups, if they represent different times, are numbered sequentially, from oldest to newest.

Attributes (Table 10c-19) may consist of a single value (timePoint) as for the gridded data, or an extended list of variables that describe several characteristics of fixed station (stationwise) time series data (dataCodingFormat = 8).

Table 10c-19 – Attributes of values groups

No	Name	Camel Case	Mult	Data Type	Remarks and/or Units
dataCodingFormat = 1, 2, 3, 4, 5, 6 or 7					
1	Time stamp	timePoint	1	String	DateTime
dataCodingFormat = 8					
1	Name of the station	stationName	0..1	String	For example, a geographic description or 'Not Available'
2	Station identification	stationIdentification	0..1	String	For example, a letter number code for the station or 'Not Available'
3	Number of records	numberOfTimes	0..1	Integer	Value here overrides the corresponding value at Instance level
4	Index for time interval	timeIntervalIndex	1	(Integer)	1 (TRUE) denotes uniform time interval; interval provided by *timeRecordInterval* 0 (FALSE) denotes non-uniform time interval. This is a Boolean data type implemented as described in Table 10c-1
5	Time interval	timeRecordInterval	0..1	Integer	Only if *timeIntervalIndex* = 1 The uniform interval between time records. Units: Seconds. Value here overrides corresponding value at Instance level
6	Valid time of earliest value	startDateTime	0..1	String	DateTime format
7	Valid time of latest value	endDateTime	0..1	String	DateTime format
	(additional attributes)				(As specified in Product Specification)

10c-9.11.1 Feature oriented Regular Grid

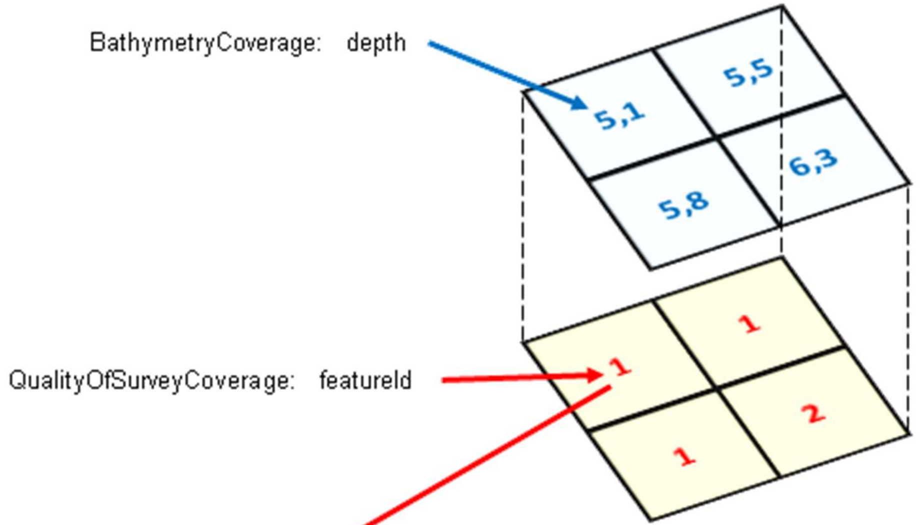
The dataCodingFormat describes a possibility to create features in form of a raster geometry and to provide them with additional information. The dataCodingFormat can only be used together with the feature attribute table (clause 10c-9.6.2, Table 10c-9).

The name of the feature container group must correspond to a camelCase name of a feature type object from the IHO Geospatial Information (GI) Registry, Data Dictionary Register.

Each grid cell defines a value that allows identification of the respective feature. The data type of the feature identification value is identical to the identification value from the feature attribute table (clause 10c-9.6.2). The code as well as the name of the feature identification value is defined in the feature information group (clause 10c-9.5); for example, index or id. The identification values of a feature may occur multiple times in the grid because multiple grid cells together determine the grid geometry of a feature. Compression of the HDF5 dataset eliminates the effect of redundancy. The feature identification value is the connection between the grid geometry in the grid and the information from the feature attribute table.

The use of this dataCodingFormat can require another Feature Container Group in raster format. This is the reference for the features created here. The spatial extent of the dataCodingFormat described here refers to the previously mentioned reference Feature Container Group. The spatial extents must match. Which Feature Cointainer Group is set as reference must be defined by the implementing Product Specification.

If no features can/should be created in a certain area of the grid, the corresponding grid cells are to be assigned the feature identification value 0 (zero). The value 0 (zero) is thus considered as nodata value. Figure 10c-11 below shows a simplified illustration using an example of S-102.



Feature Attributes

id	surveyAuthority	surveyDateRange.dateStart	techniqueOfVerticalMeasurement	...
1	BSH	20210216	2:foundByMultiBeam	
2	WSA HH	20180723	1:foundByEchoSounder	
...				

Figure 10c-11 – Illustrative example of the dataCodingFormat

Page intentionally left blank

10c-10 Common enumerations and dictionaries

10c-10.1 Data coding format

Table 10c-20 – Data coding format enumeration

Item	Name	Description	Code	Remarks
Enumeration	S100_HDF_DataCodingFormat	Data coding formats for S-100 HDF5 data		
Literal	fixedStations	Data at multiple discrete fixed point locations	1	
Literal	regularGrid	Data at grid points forming a regular grid with constant cell spacing	2	Regular grids are commonly composed of perpendicularly crossing lines of equal spacing on each dimension, creating square or rectangular cells
Literal	ungeorectifiedGrid	Data that does not include any information that can be used to determine a cell's geographic coordinate values, or in which cell spacing is variable, and there is no predefined association between one cell's location and that of another	3	For example, a digital perspective aerial photograph without georectification information included
Literal	movingPlatform	Data at sequential discrete point locations of a moving sensor platform	4	
Literal	irregularGrid	Data distributed over a grid with uniform cell spacing but irregular overall shape	5	The irregularity of shape may consist of non-rectangular coverage area or relatively large regions which are not populated with data
Literal	variableCellSize	Variable-density grid containing one or more regions with cell spacing that is a whole multiple of a common minimum uniform cell spacing	6	The shape of the overall grid may be non-rectangular
Literal	TIN	Triangulated irregular network	7	A TIN is a representation of a continuous surface consisting entirely of triangular facets. The vertices at the corners of each triangle are shared with the adjacent triangle. These vertices form the control points of the coverage function
Literal	stationwiseFixed	Data at multiple discrete fixed point locations organised by station	8	
Literal	featureOrientedRegularGrid	Data at grid points forming a regular grid with constant cell spacing	9	Regular grids are commonly composed of perpendicularly crossing lines of equal spacing on each dimension, creating square or rectangular cells

10c-10.2 Type of the horizontal CRS

Table 10c-21 – Type of the horizontal CRS

Item	Name	Description	Code	Remarks
Enumeration	typeOfHorizontalCRS	Codes for describing the type of the two-dimensional horizontal CRS		
Literal	geodeticCRS2D	Two-dimensional geodetic CRS	1	
Literal	projectedCRS	Projected CRS	2	

10c-10.3 Vertical coordinate base

Table 10c-22 – Vertical coordinate base

Item	Name	Description	Code	Remarks
Enumeration	verticalCoordinateBase	Codes for describing the base level of the vertical coordinate system		
Literal	seaSurface	The base of the vertical coordinate system is the sea surface	1	
Literal	verticalDatum	The base of the vertical coordinate system is a defined vertical datum	2	
Literal	seaBottom	The base of the vertical coordinate system is the sea floor	3	

10c-10.4 Vertical datum reference

Table 10c-23 – Vertical datum reference

Item	Name	Description	Code	Remarks
Enumeration	verticalDatumReference			
Literal	s100VerticalDatum	The vertical datum is one of those listed in S100_VerticalAndSoundingDatum	1	See clause 10c-10.6
Literal	EPSG	The vertical datum is one of those listed in the EPSG Registry	2	

10c-10.5 Projection methods

Table 10c-24 – Projection methods and their parameters

Name	EPSG Code	Parameter 1	Parameter 2	Parameter 3	Parameter 4	Parameter 5
Mercator	9805	Latitude of 1st standard parallel	Longitude of natural origin	-	-	-
Transverse Mercator	9807	Latitude of natural origin	Longitude of natural origin	Scale factor at natural origin	-	-
Oblique Mercator	9815	Latitude of projection centre	Longitude of projection centre	Azimuth of initial line	Angle from Rectified to Skew Grid	Scale factor on initial line
Hotline Oblique Mercator	9812	Latitude of projection centre	Longitude of projection centre	Azimuth of initial line	Angle from Rectified to Skew Grid	Scale factor on initial line

Lambert Conic Conformal (1SP)	9801	Latitude of natural origin	Longitude of natural origin	Scale factor at natural origin	-	-
Lambert Conic Conformal (2SP)	9802	Latitude of false origin	Longitude of false origin	Latitude of 1st standard parallel[4]	Latitude of 2nd standard parallel[5]	-
Oblique Stereographic	9809	Latitude of natural origin	Longitude of natural origin	Scale factor at natural origin	-	-
Polar Stereographic	9810	Latitude of natural origin[6]	Longitude of natural origin	Scale factor at natural origin	-	-
Krovak Oblique Conic Conformal	9819	Latitude of projection centre	Longitude of projection centre	Azimuth of initial line	Latitude of pseudo standard parallel	Scale factor on pseudo standard parallel
American Polyconic	9818	Latitude of natural origin	Longitude of natural origin	-	-	-
Albers Equal Area	9822	Latitude of false origin	Longitude of false origin	Latitude of 1st standard parallel[5]	Latitude of 2nd standard parallel[6]	-
Lambert Azimuthal Equal Area	9820	Latitude of natural origin	Longitude of natural origin	-	-	-

NOTE: All latitudes and longitudes of the projection parameters must be given in degrees (south and west negative). Azimuths are given in degrees. For detailed description of the projection method refer to the EPSG documentation.

10c-10.6 Vertical and sounding datum

Table 10c-25 – Vertical and sounding datum

Item	Name	Description	Code	Remarks
S100_Codelist	S100_VerticalAndSoundingDatum	Allowable vertical and sounding datums	-	Open enumeration
Value	meanLowWaterSprings		1	(MLWS)
Value	meanLowerLowWaterSprings		2	-
Value	meanSeaLevel		3	(MSL)
Value	lowestLowWater		4	-
Value	meanLowWater		5	(MLW)
Value	lowestLowWaterSprings		6	-
Value	approximateMeanLowWaterSprings		7	-
Value	indianSpringLowWater		8	-
Value	lowWaterSprings		9	-
Value	approximateLowestAstronomicalTide		10	-
Value	nearlyLowestLowWater		11	-
Value	meanLowerLowWater		12	(MLLW)
Value	lowWater		13	(LW)
Value	approximateMeanLowWater		14	-

[4] Standard parallel nearer equator.

[5] Standard parallel farther from equator.

[6] Must be either 90 degrees or -90 degrees.

Value	approximateMeanLowerLowWater		15	-
Value	meanHighWater		16	(MHW)
Value	meanHighWaterSprings		17	(MHWS)
Value	highWater		18	(HW)
Value	approximateMeanSeaLevel		19	-
Value	highWaterSprings		20	-
Value	meanHigherHighWater		21	(MHHW)
Value	equinoctialSpringLowWater		22	-
Value	lowestAstronomicalTide		23	(LAT)
Value	localDatum		24	-
Value	internationalGreatLakesDatum1985		25	-
Value	meanWaterLevel		26	-
Value	lowerLowWaterLargeTide		27	-
Value	higherHighWaterLargeTide		28	-
Value	nearlyHighestHighWater		29	-
Value	highestAstronomicalTide		30	(HAT)
Value	balticSeaChartDatum2000	Baltic Sea Chart Datum 2000	44	-
Value	internationalGreatLakesDatum2020	The 2020 update to the International Great Lakes Datum (IGLD), the official reference system used to measure water level heights in the Great Lakes, connecting channels, and the St. Lawrence River system	46	Unlike the previous two IGLDs, this datum update will use a geoid-based vertical datum that will be accessible using Global Navigation Satellite Systems (GNSS) such as the Global Positioning System (GPS)
Value	seaFloor	The bottom of the ocean and seas where there is a generally smooth gentle gradient. Also referred to as sea bed (sometimes seabed or sea-bed), and sea bottom	47	-
Value	seaSurface	A two-dimensional (in the horizontal plane) field representing the air-sea interface, with high-frequency fluctuations such as wind waves and swell, but not astronomical tides, filtered out	48	-
Value	hydrographicZero	A vertical reference near the lowest astronomical tide (LAT), below which the sea level falls only very exceptionally	49	Deviation between LAT and hydrographic zero may be due to a strong anticyclonic atmospheric condition, adding weight to the water column that may exceptionally cause the lowest sea level to fall below the astronomical low water level

NOTE: The numeric codes are the codes specified in the IHO GI Registry for the equivalent listed values of the IHO Hydro domain attribute *Vertical Datum,* since the Registry does not at present (20 June 2018) contain entries for Exchange Set metadata and dataset metadata attributes.

Datums not included in the S-100 enumeration must be encoded using the “other: …” form. If the datum in question is listed in the IHO GI Registry (as one of the standard listed values for attribute *Vertical Datum* in the IHO Hydro domain), the “camel case code” in the Registry must be used in the “other: …” element. For datums from the EPSG Registry but not listed in the IHO GI Registry, the form should be “other: EPSG_NNNN”.

EXAMPLE 1: “Local Low Water Reference Level” is in the IHO GI Registry but not listed in the S-100 standard. It must be encoded with the camel case in the GI registry as: “other: localLowWaterReferenceLevel”.

EXAMPLE 2: “European Vertical Reference Frame 2019 mean tide” is in the EPSG Registry list of vertical datums (EPSG 1287) but not in the IHO GI Registry list. It must be encoded as: “other: EPSG_1287”.

If the datum is not listed in any the table above, the IHO GI Registry, or the EPSG Registry, producers should determine a suitable special code in consultation with the IHO Working Group(s) and the IHO GI Registry authority.

The use of datums that are neither in the enumeration above, nor in the IHO GI Registry, nor the EPSG Registry is discouraged. Producers who need to use a datum not listed in the S-100 enumeration should propose its addition to the IHO GI Registry and/or this enumeration by means of an S-100 maintenance proposal.

Note that application software is not required to process information encoded in “other: …” form, meaning that ECDIS software, for example, is not required to recognise any datum encoded as “other: …” and will therefore be unable to adjust ENC depth information with water level data from the corresponding S-104 dataset, and may warn or reject the S-104 dataset as being incompatible with S-101 ENCs.

10c-11 Support files

The HDF5 format does not encode support file information as feature attributes; that is, application schema thematic attributes cannot be references to support files. This means that references to pictures or text files, etc, are not permitted in coverage features.

Also, feature and information associations from coverage to vector features are not permitted.

The HDF5 “metadata” attribute of the root group is a reference to an external metadata file. The reference must be a string of the form:

fileRef:<fileName>

where <fileName> is the base name of the ISO 8211 or GML file. The extension part of the file name is not used.

Mixed vector-coverage data products may continue to use support files in connection with vector feature classes and define vector feature or information classes with attributes that are references to support files, as usual.

10c-12 Catalogue and metadata files

Exchange set catalogues and metadata files must conform to the standard XML schemas for catalogues and metadata defined for this edition of S-100 and the relevant ISO standards. The files must be named as follows:

CATALOG.XML (or .xml) Exchange catalogue XML file.

MD_<HDF5 data file base name>.XML (or .xml) ISO metadata

10c-13 Vector spatial objects, features, and information types

In some circumstances it may be necessary to use vector spatial objects, such as area of influence polygons. This edition of the profile does not encode vector spatial objects directly in the HDF5 data file. Instead, the spatial objects should be defined in an external file (either GML or ISO 8211 format) and a reference to the spatial object encoded. The reference must be a string of the form:

extObjRef:<fileName>:<recordIdentifier>

where <fileName> is the base name of the ISO 8211 or GML file, and <recordIdentifier> is the record identifier of the vector object record within that file. The extension part of the file name is not used. The record identifier is the gml:id for GML datasets, or the record identification number (RCID) for ISO 8211 datasets. The file must be present in the same exchange set.

This method can be used to reference polygons, etc, defined in external files in GML or 8211 format data files in the same exchange set. It can also be used to reference feature or information type instances in the GML or ISO 8211 file.

EXAMPLES:

USSFC00001:S093546 references the object with gml:id S093456 in the GML data file USSFC00001.GML (GML).

USSFC00001:93546 references the object with record identifier 93456 in the ISO 8211 data file USSFC0000.000 (ISO 8211).

10c-14 Constraints and validation

10c-14.1 Validation tests

Validation tests must be defined in the Product Specification, and include checks that:

- HDF5 file structure conforms to this profile;
- Mandatory attributes in the groups are present according to the encoded value of dataCodingFormat;
- Group, dataset, and attribute names conform to this profile;
- Lengths of positioning and value records arrays are consistent;
- Components of compound types are named as required by the specification.

10c-15 Updates

Updates to HDF5 datafiles are recommended to follow the same structure as the base HDF5 datafile. Updates may include only the HDF5 datasets which are being updated. The specific datasets being updated are included in their entirety in the update datafile.

This clause implies that S-100 datasets may be updated in part as well as replaced completely by updated data, but Product Specifications are not required to permit partial updates. They may define update creation and management processes which are more suitable for their particular domains and applications. However, if updates to parts of S-100 datasets are allowed, the rule in the previous paragraph must be followed.

10c-16 Summary of model

The basic structure of the HDF5 profile (Figure 10c-7) can now be presented as a more detailed conceptual model using the group and dataset specifications in the previous sections. The conceptual model of HDF5 file contents is shown in the following Figure. This Fgure shows the group structure and the datasets which contain spatial representations and data values. (Metadata attributes and datasets containing metadata are not included for the sake of simplicity.) The *MatchingOrders* association indicates that the sequences of elements in the associated datasets are interdependent.

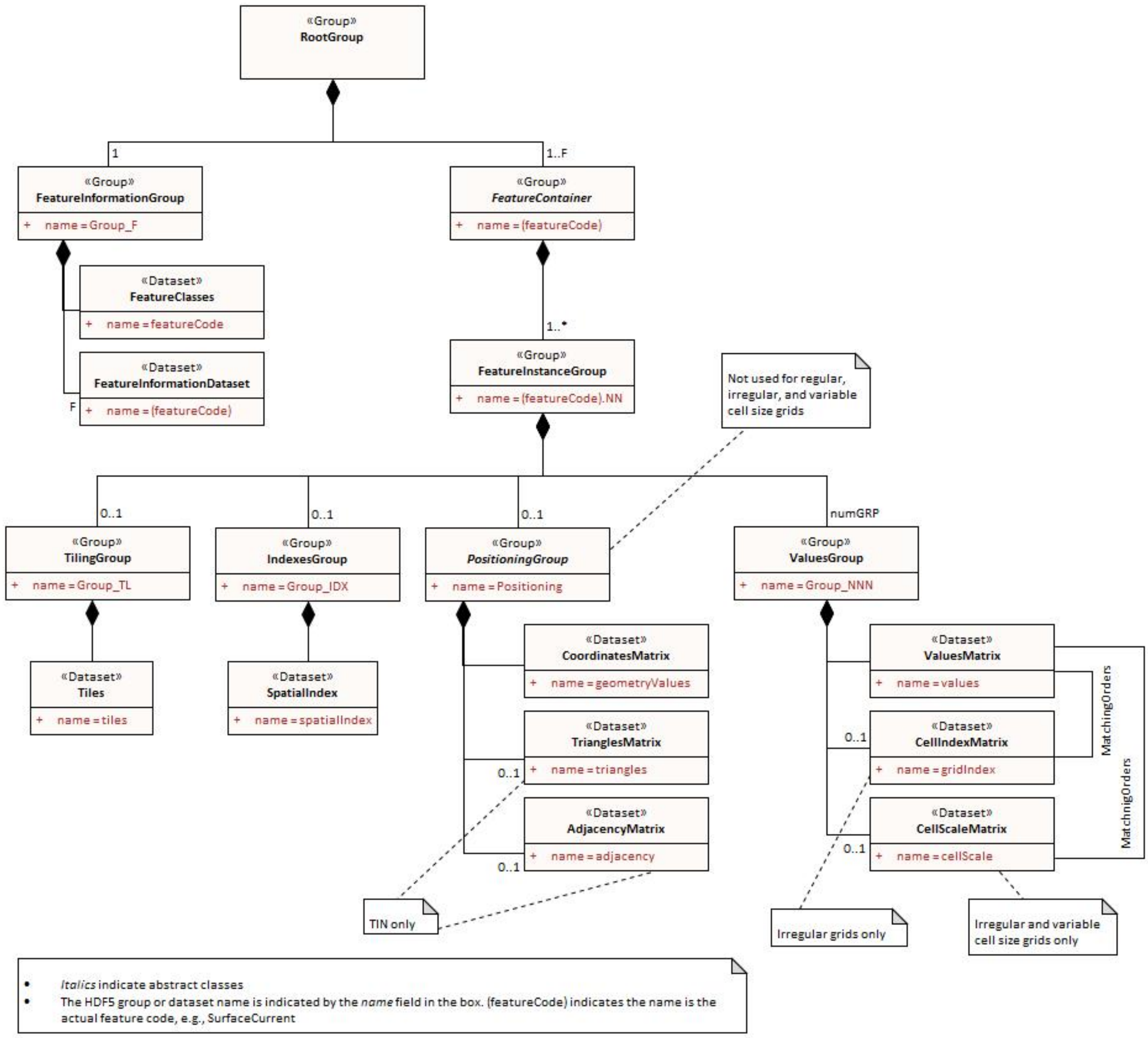


Figure 10c-12 - Conceptual model of content

10c-17 Rules for Product Specification developers

10c-17.1 Defining the format for a Product Specification from this profile

Most Product Specifications will need only a subset of this profile. However, all Product Specifications must include the mandatory elements of this profile.

The logical structure of the datafile must conform to the logical structure depicted in Figure 10c-11 and specified in the preceding sections.

The 'Data Format' section of the Product Specification must indicate what part of the profile is used (for example, which values *dataCodingFormat* can take, which groups and datasets are used, whether the spatial representation is 2-dimensional, 3-dimensional, etc).

UML diagrams derived from the conceptual structure depictions in this Part are recommended but not mandatory. Documentation tables specifying product-specific constraints or limitations on metadata and content must be provided unless the corresponding table in this profile applies without modification.

Specifications which require grids with non-uniform spacing must be treated as ungeorectified grids and have the coordinates of each position explicitly encoded.

This profile does not prevent a feature class from having different coverage types of coverage, but repeating spatial attributes for the same instance is not possible in this profile. This means that a feature instance cannot have two grids, whether or not they are the same coverage type. If Product Specifications appear to need multiple coverages for the same instance, consider combining the two into a single coverage object or using two feature instances.

Feature and information associations are not fully implemented in this profile. However, it is possible to link coverage objects to vector feature or information objects in accompanying GML or ISO 8211 datasets using the object reference methods described in clause 10c-13. References to vector objects, such as influence polygons must be encoded using the same method.

Product Specifications should specify the precision of the numeric metadata elements which are encoded in the HDF5 datafile, either individually or in blanket statements. For example, a Product Specification may require that all the metadata attributes of type Float be encoded using 64-bit floating point numbers.

If uncertainty in positions or data values varies over the spatial extent of a single feature, Product Specification developers should consider solutions as part of the Product Specification; for example, subdividing the grid into different feature instances, or addressing this at the application schema level by defining an overlay feature to encode uncertainties or adding an uncertainty attribute to the values record. This Part does not require any specific approach to this problem.

10c-17.2 Miscellaneous rules

The use of variable length strings as components of compound types is discouraged due to reported performance problems.

In theory, the use of tiles can interact with HDF5 chunking to affect performance. Product Specifications for which performance is a significant consideration may need to consider possible interaction effects and investigate their magnitude and consequences.

10c-17.3 Extensions of this profile

Product Specifications may extend the format in this profile by defining additional data structures or extending the data structures defined in this profile, but all extensions must retain the core specifications of this profile so that **implementations must be able to ingest and portray data without processing the additional data structures**. **The Product Specification must be written so that use of these extra data structures for processing or portrayal is optional.**

Such additions should be placed in the appropriate location in the HDF5 data file; for example, spatial indexes in the Group_IDX group.

Extensions must not reuse the names of items defined in this profile. Items defined in this profile must not be renamed in Product Specifications.

Some examples of permissible and impermissible extensions are given below.

- Permissible extensions:
  - Quadtree index, added as an HDF5 dataset in the indexes group.
  - Extension of the value record structure that retain the core format described in this profile (that is, the 1-d array structure and the specified components).
  - Linear scale arrays indicating the grid points on each axis where the cell size changes, as an adjunct to variable cell size arrays.
  - Product-specific metadata as attributes of any of the groups specified in this profile.
  - Product-specific metadata as additional HDF5 datasets in any of the groups specified in this profile.
  - Additional groups, provided these are not used as substitutes for one of the mandatory groups in this profile.
- Impermissible extensions:
  - Changes to the rank of an array dataset type; for example, using a 2-d array in place of a 1-d array.

- Changes to the rules for naming of a component of a compound data type defined in this profile.

10c-17.4 Extensions that add metadata

While clause 10c-17.3 permits adding metadata, defining product-specific metadata means that implementation must – if they are to do anything with the additional metadata other than merely display it – include product-specific coding in applications. Given that the S-100 ecosystem includes multiple data products which would ideally all be processable (including portrayal) by an S-100 application, this Part recommends against adding product-specific metadata that has any effects on processing or portrayal. If such additions are considered essential they should be proposed as an extension to the S-100 framework itself using the maintenance mechanism described in S-100 and related documents. Display-only metadata (that is, where the application is only expected to display the content of the added attribute) may be added but is discouraged.

10c-18 Implementation guidance

The HDF5 C API includes interfaces for determining the types of compound type components. This suggests that the size of a datatype can be checked to mitigate possible conversion issues.

The HDF5 C API also defines iterators for iterating over attributes or items in a group. These iterators can be used to discover profile datasets, groups, or attributes from datasets, groups, and attributes defined only in individual Product Specifications (the product-specific items will have names different from the profile items).

The order in which objects are retrieved may not be the same as the creation order. Implementers should allow for this or investigate the availability of order-preserving functions in the HDF5 API.

Linkage between the XML feature catalogue and objects in the HDF5 file is preserved by using the (camel case) codes for features, and attributes.

10c-19 Application Schema Realizations from Part 8 (Informative)

Figure 10c-13 below illustrates the realization of different types of coverages as described in clause 10c-10.1 and Table 10c-4 from the coverage types defined in Part 8. Product Specification development teams may wish to develop similar figures for their Product Specifications to describe how the coverage types as encoded in the HDF5 format for their data products are realized from the coverage types defined in Part 8.

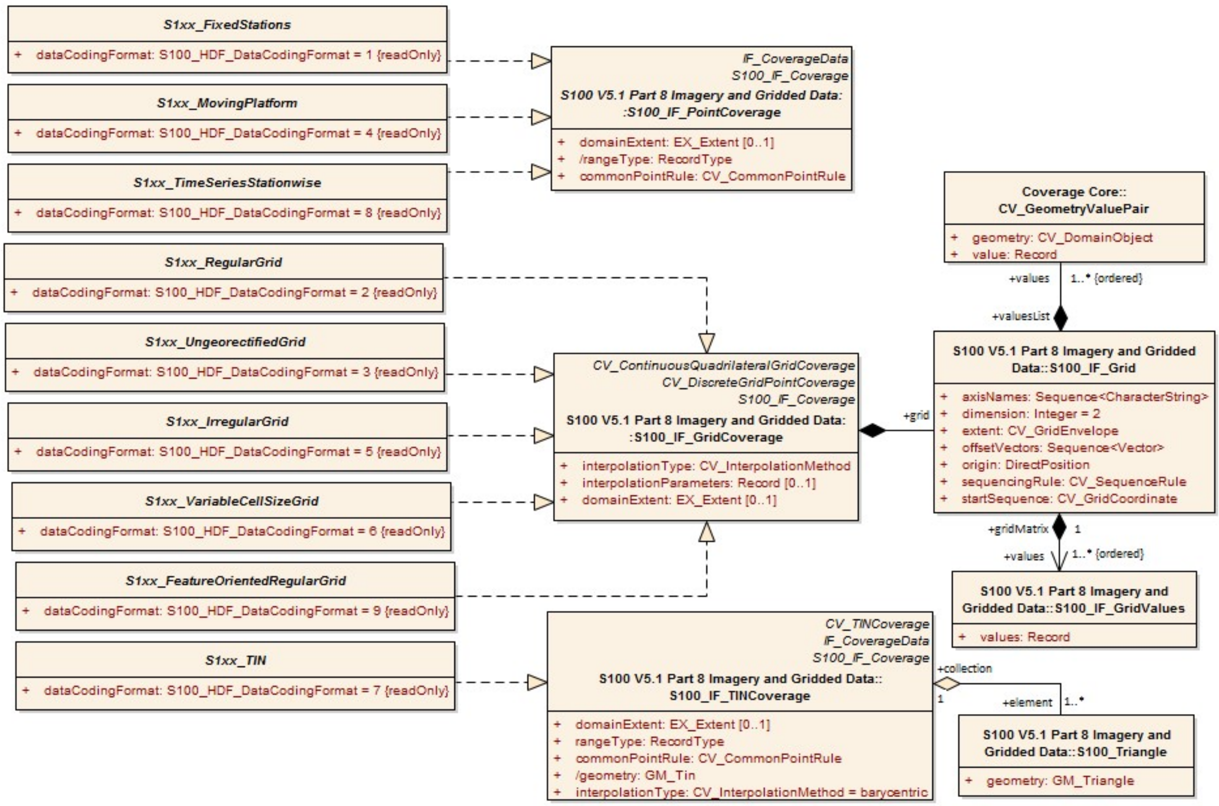


Figure 10c-13 - Realizations of HDF5 encoding coverage types from Part 8 types

It is possible to extend Figure 10c-13 to include upper-level derivations from ISO types, at the potential expense of a more crowded figure. Figure 10c-14 illustrates the idea for the Regular Grid coverage type. Note that the hypothetical Application Schema allows both discrete grid point coverage and continuous grid coverage types; if only continuous grid coverage is needed the CV_DiscreteGridPointCoverage and CV_DiscreteCoverage classes can be omitted. Also, since this hypothetical Application Schema uses only regular grids, the coordinates of grid points can be calculated from the grid origin and cell size, and the CV_GeometryPair class associated to S100_IF_Grid is not needed and has been omitted from Figure 10c-14.

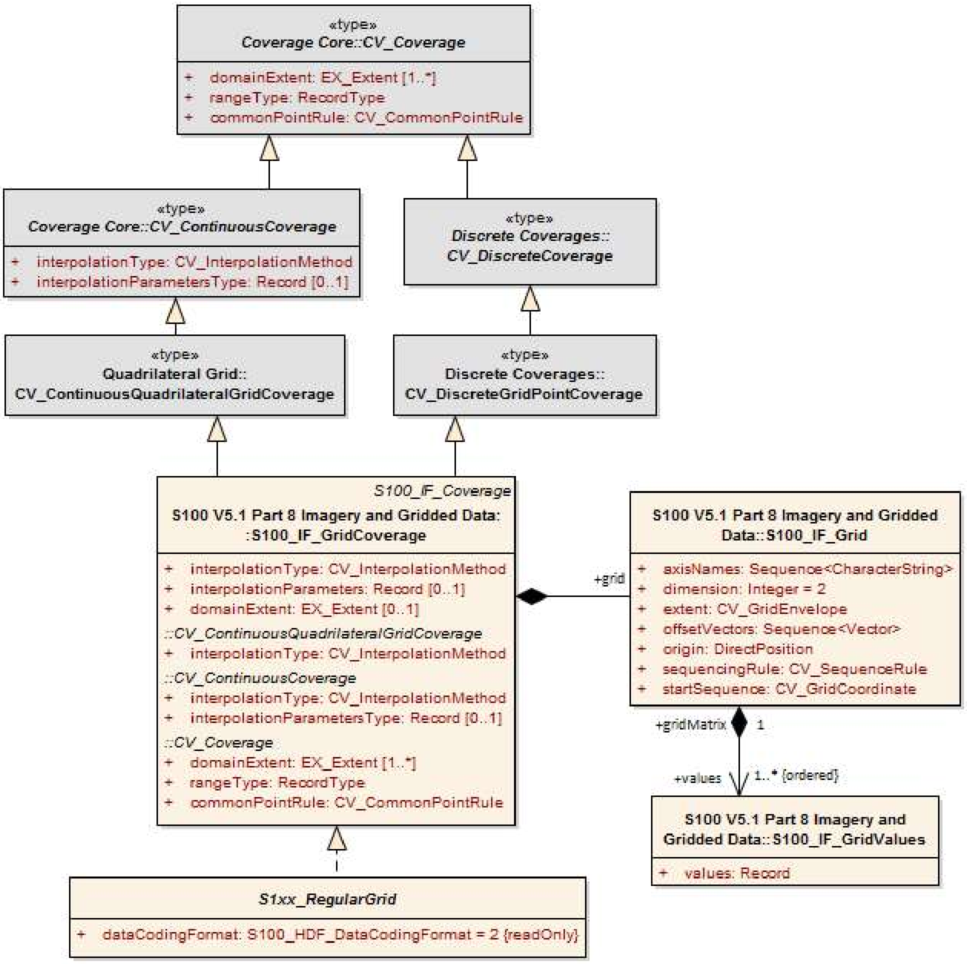


Figure 10c-14 - Hypothetical Application Schema fragment including upper-level ISO types. The grey boxes are classes defined in ISO 19123

Page intentionally left blank

S-100 – Part 11

Product Specifications

Page intentionally left blank

Contents

Page intentionally left blank

11-1 Scope

A data product specification is a precise technical description which defines a geospatial data product. It describes all the features, attributes and relationships of a given application and their mapping to a dataset. It includes general information for data identification as well as information for data content and structure, reference system, data quality aspects, data capture, maintenance, delivery and metadata. It may be created and used on different occasions, by different parties and for different reasons.

This Part of S-100 describes data product specifications for hydrographic requirements for geographic data products. Its aim is to provide a clear and similar structure for any data product specification to be written. This profile shall be in conformance with all the other standards that have been developed within the IHO S-100 Geospatial Standard for Hydrographic Data.

The product specification shall constitute a set of human readable documentation. Generally, it should also include machine readable files for information such as the feature catalogue, the application schema and the CRS parameters. An example of a compliant product specification is shown in Appendix 11-B.

In addition to a 'human readable' document, it is possible to create a machine readable (for example XML) summary of the Product Specification. The tables in the sections below indicate the structure for such a summary of the Product Specification.

11-2 References

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

11-2.1 Normative

ISO 639-2:1998, *Codes for the representation of names of languages – Part 2: Alpha-3 code*

ISO 19115-1:2018, *Geographic information – Metadata – Part 1 – Fundamentals* (2014) (as amended by Amendment 1, 2018)

ISO 19131:2007, *Geographic information – Data product specifications*

ISO 19157:2018, *Geographic information -- Data qualit*y (2013) (as amended by Amendment 1, 2018)

11-2.2 Informative

ISO 8211:1994, *Information technology — Specification for a data descriptive file for information interchange*

ISO 19104:2004, *Geographic information – Terminology*

ISO 19106:2004, *Geographic information – Profiles*

ISO 19109:2005, *Geographic information – Rules for application schema*

ISO 19115:2003, *Geographic information – Metadata*

ISO 19123, *Geographic information – Schema for Coverage Geometry and Functions*

ISO 19136, *Geographic information – Geography Markup Language*

ISO 19138, *Geographic information – Data quality measures*

11-3 General structure and content of a data product specification

A data product specification defines the requirements for a data product and forms the basis for producing or acquiring data. The data product specification shall contain sections covering the following aspects of the data product:

a) Overview – see Clause 11-4;
b) Specification scopes – see Clause 11-5;
c) Data product identification – see Clause 11-6;
d) Data content and structure – see Clause 11-7;
e) Reference systems – see Clause 11-7.3;
f) Data quality – see Clause 11-8;
g) Data Capture – see Clause 11-9

NOTE This section can be covered by an encoding guide, for example for the S-101 ENC Product Specification - the Data Classification and Encoding Guide;

h) Data product format – see Clause 11-12;
i) Data product delivery – see Clause 11-13;
j) Metadata – see Clause 11-15.

A data product specification may also contain sections covering the following aspects of the data product:

k) Data Maintenance – see Clause 11-10;
l) Portrayal – see Clause 11-11;
m) Additional Information – see Clause 11-14.

Each of these sections of the data product specification is described in the following clauses.

NOTE Sections are adopted from ISO 19131

11-4 Overview

The overview section provides a reader of a data product specification with general introductory information about the data product together with product specification metadata.

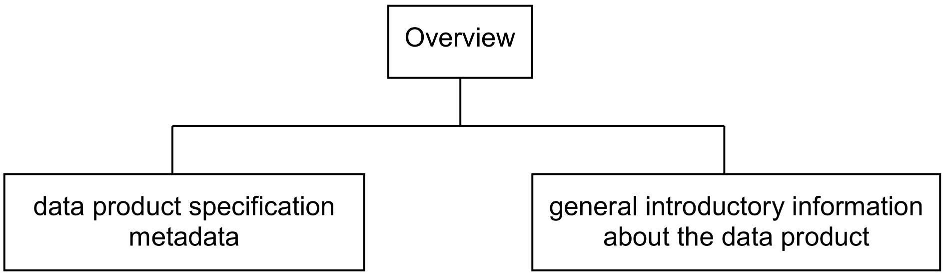


Figure 11-1 Content of the Overview Section

The Overview shall include the following parts:

a) information about the creation of the data product specification;

NOTE This shall include the title, a reference date, the responsible party and the language. Information about the maintenance regime for the product specification should also be included;

b) terms and definitions;
c) abbreviations;
d) acronyms for the name of the data product

EXAMPLE AML Additional Military Layer;

e) an informal description of the data product.

The information shall contain general information about the data product which may include the following aspects shown in Table 11-1.

Table 11-1— Informal Description of the Data Product

Name	Description	Mult	Type
title	Official designation of the data product	1	CharacterString
abstract	Informal description of the data product	1	CharacterString
acronym	Any acronyms for the title of the data product	0..*	CharacterString
content	Textual description of the content of any dataset which conform to the specification	1	CharacterString
spatialExtent	Description of the spatial extent covered by the data product	1	EX_Extent (ISO 19115-1)
temporalExtent	Description of the temporal extent covered by the data product	0..1	EX_Extent (ISO 19115-1)
specificPurpose	Specific purpose for which the data shall be or has been collected	1	CharacterString

The data product specification metadata shall provide information to uniquely identify the data product specification as well as information about the creation and maintenance of the data product specification. The maintenance description may indicate regular updates, or give contact details for reporting issues which need correction. The data product specification metadata shall include the following items in Table 11-2 [extension to ISO 19131].

Table 11-2— Data product specification metadata

Name	Description	Mult	Type
title	Title of the data Product Specification	1	CharacterString
version	Version of the data Product Specification	1	CharacterString
date	Date the Product Specification was created / last updated	1	Date
language	Language(s) of the data Product Specification, for example translations	1..*	CharacterString
classification	Security classification code on the data Product Specification	0..1	MD_ClassificationCode (ISO 19115-1)
contact	Party responsible for the data Product Specification	1	CI_Responsibility (ISO 19115-1)
URL	Online-address where the resource is downloadable	0..1	URL
identifier	Persistent unique identifier for a published version of the Product Specification[1]	1	CharacterString
maintenance	Description of the maintenance regime for the Product Specification	1	MD_MaintenanceInformation (ISO 19115-1)
compliancyCategory	The level of compliance of the Product Specification to S-100	0..1	S100_compliancyCategory

11-5 Specification scopes

Some parts of a product specification may apply to the whole product whereas other parts of the product specification may apply to parts of the product. Coordinate reference system will generally apply to the complete product; whereas maintenance regimes may be different for navigational features and contextual features. If a specification is homogeneous across the whole data product it is only necessary to define a general scope (root scope), to which each section of the data product specification applies. The data product specification may specify a partitioning of the data content of the product on the basis of one or more criteria. Such partitioning may be different for different parts of the data product specification. Each such part of the data content shall be described by a specification scope that may inherit or override the general scope specification.

[1] This is referenced from the discovery metadata of products which conform to the Product Specification

In principle, any or all of the remaining sections of the product specification may have variants which apply to the scopes within the product. Each variant must identify the scope(s) to which it applies.

EXAMPLE Data products to support navigation often contain two sets of feature types: those that provide navigation information that changes rapidly and is essential for safety of navigation, and those that provide background reference information. Maintenance and delivery information would be partitioned on the basis of these groupings; reference system information would not.

This section is only used where different parts of the product (e.g. by theme or geographical extent) have different specifications. For example, some aspects of the specification may be specific to bathymetry, or to non-tidal waters. If this is the case for the product being specified, this section defines the various "scopes" within the overall product specification, and how they should be identified in the datasets.

Depending on the type of data product specification the scope may include items in Table 11-3.

Table 11-3— Specification Scope Information

Name	Description	Mult	Type
scopeIdentification	Specific identification of the scope	1	CharacterString
level	Hierarchical level of the data specified by the scope	0..1	MD_ScopeCode (ISO 19115-1)
levelName	Name of the hierarchy level	0..1	CharacterString
levelDescription	Detailed description about the level of the data specified by the scope	0..1	CharacterString
coverage	Subtype of a feature that represents real world phenomena as a set of attributes	0..1	CharacterString
extent	Spatial, vertical and temporal extent of the data	0..1	EX_Extent (ISO 19115-1)

11-6 Data product identification

This section describes how to identify data sets that conform to the specification. The information identifying the data product may include the following items in Table 11-4. [adopted from ISO 19131].

Table 11-4— Identification Information

Name	Description	Mult	Type
title	The title of the data product	1	CharacterString
alternateTitle	Short name or other name by which the data product is known	0..1	CharacterString
abstract	Brief narrative summary of the content of the data product	1	CharacterString
topicCategory	The main theme(s) of the data product	0..*	MD_TopicCategoryCode (ISO 19115-1)
geographicDescription	Description of the geographic area covered by the data product using identifiers	1	EX_GeographicDescription (ISO 19115-1)
spatialResolution	Factor which provides a general understanding of the density of spatial data in the data product	1	MD_Resolution (ISO 19115-1)
purpose	Summary of the intention with which the data product is developed	1	CharacterString
language	Language(s) of the dataset. If language is not applicable, for example for raster data, use "not applicable" as value for the element	1..*	CharacterString (ISO 639-2)
classification	Security classification code on the data product	0..1	MD_ClassificationCode (ISO 19115-1)
spatialRepresentationType	Form of the spatial representation	0..1	MD_SpatialRepresentation TypeCode (ISO 19115-1)

pointOfContact	Identification of, and means of communication with, person(s) and organization(s) associated with the data	0..*	CI_Responsibility (ISO 19115-1)
useLimitation	Limitation affecting the fitness for use of the data product	0..1	CharacterString

11-7 Data content and structure

This profile mandates different requirements for data product specifications whether the data is feature- or coverage-based or imagery data. The product specification shall include this information for each identified scope.

11-7.1 Feature-based data

The content information of a feature-based data product is described in terms of a general feature model and a Feature Catalogue [adopted from S-100 Part 3 and S-100 Part 5].

The data product specification shall contain an application schema. For all data product specifications in the realm of S-100, the application schema shall be expressed in UML. All other rules of S-100 Part 3 concerning the creation of the general feature model and especially conformance to ISO 19109:2005 apply as well. If the application schema is a separate document, then the product specification shall include a narrative summary. The product specification shall describe any fixed roles or other restrictions or conventions for default roles. If unique role names are required, it may also define conventions for generating these unique names.

The data product specification shall include a feature catalogue, which provides a full description of each feature type including attributes, attribute values and relationships in the data product. The feature catalogue shall be realized in accordance with S-100 Part 5. The feature catalogue shall be available in both 'machine readable' (for example XML based on the S-100 Feature Catalogue XSD) and 'human readable' (for example textual derived by XSLT from the XML) forms.

All the feature types, their attributes and attribute value domains, and the association types between feature types expressed in the application schema shall be described in a feature catalogue.

The Product Specification for feature-based scopes shall include the elements in Table 11-5.

Table 11-5 — Elements of Feature-based data

Name	Description	Mult	Type
applicationSchema	The application schema	1	DPS_ApplicationSchema
featureCatalogue	The feature catalogue	1	FC_FeatureCatalogue

11-7.2 Coverage-based and imagery data

The content information of a coverage-based data product (including imagery data product) shall be described in accordance with S-100 Part 7. The content information shall be described in the following manner:

A data product specification shall identify each coverage type and each image type that is included within the specification scope and shall provide a narrative description for each. Accordingly, the following components shall be identified to describe a coverage or an image (Table 11-6):

Table 11-6— Coverage-based and imagery data

Name	Description	Mult	Type
coverageID	Unique identifier of coverage	1	CharacterString
coverageDescription	Technical description of the coverage	1	CharacterString
coverageType	Type of the coverage	1	CharacterString
specification	Additional information	1	CV_Coverage (ISO 19123)

11-7.3 Coordinate Reference Systems

The data product specification shall include information that defines the reference systems used in the data product. The spatial reference system used shall be a coordinate reference system (CRS) in conformance with S-100 Part 6 CRS Component. The application schema will show how CRS references are carried in the data sets; this may be by reference to a register of CRS parameters, such as the EPSG Geodetic Parameter Dataset.

A product specification may express coordinate operation parameters for operations between particular CRSs. These parameters shall be recorded as described in S-100 Part 6.

Table 11-7— Reference system identification

Name	Description	Mult	Type
spatialReferenceSystem	Reference system identifier(s) of spatial reference system used, for example different UTM zones can be considered as different reference systems	1..*	SC_CRS (S-100 Part 6)

11-7.4 Object identifiers

The specification of persistent global identifiers for feature and information objects is strongly recommended. Identifiers need not be defined where the physical realities dictate otherwise or it is known that a reference to the object will not be needed, even from an as-yet-unknown external dataset conforming to another product specification. For example, identifiers need not be defined for cartographic objects.

Identifiers of instances should utilize the Maritime Resource Name (MRN) concept and namespace. The MRN namespace is administered by International Association of Lighthouse Authorities (IALA) through the website http://mrnregistry.org, which also contains references to the full set of rules that apply to the MRN concept. The topmost namespace urn:mrn remains fixed, with subsequent name spaces separated by colons, and available through the application process explained on the website. Any organization wishing to issue MRN conformant identifiers should apply for a name space from IALA, or from an organization that already has a namespace registered.

It is not required to encode all feature instances with the whole MRN string, provided the whole string can be recreated, for example by utilizing metadata. Significant data volume savings can be obtained by utilizing such mechanisms. Furthermore, technical issues such as restrictions in GML encoding with the use of ":", may be surmounted by this approach.

If there are technical reasons why the MRN concept cannot be utilized, other means for persistent global identifiers should be established. One way to implement persistent global identifiers is by defining a namespace and a persistent unique local identifier for individual feature or information types. The persistent global identifier can be constructed by combining the namespace with the local identifier. Local identifiers must be unique within the namespace for the lifetime of the feature or information object.

The local identifier must be an attribute of feature and information data objects whenever it is defined. The persistent global identifier need not be a data object attribute if the namespace portion can be computed from metadata.

Namespaces may be specified by construction, for example a rule describing how to construct a namespace from available metadata. Product Specifications must specify how persistent global identifiers are to be constructed from namespace and local identifiers.

Product Specifications should note that location-based identifiers may not be sufficient to disambiguate data objects, because (for example) two agencies might issue AtoNs in the same area, for example physical buoys marking a channel and a virtual AtoN marking section of the channel with low air draft. Updating and normalizing the data in this case must take into consideration that the two items have similar characteristics (location, aids to navigation, etc), but are different items. Therefore a location based identifier is likely not enough to enable a link between data.

11-8 Data Quality

The data product specification shall identify the data quality requirements for each scope within the data product in accordance with S-100 Part 4c. For every data quality scope it is necessary to list all the data quality elements and data quality sub-elements defined in S-100 Part 4c, even if only to state that a specific data quality element or data quality sub-element is not applicable for this data quality scope.

Each product specification shall describe the data quality requirements. One aspect is the "data quality overview element" which should allow a user to decide whether this dataset is the one they want. The other aspect is the metadata allowed for specific feature collections, features and attributes within the dataset.

The data quality overview element should include at least the intended purpose and statement of quality or lineage. Other data quality elements cover: completeness, logical consistency, positional accuracy, temporal accuracy, thematic accuracy, and anything specifically required for the product being specified.

The product specification should comment on which of these are to be used and how, including a description of (or reference to) conformance tests. For example, should data only be published if it passes a particular test, or is it allowable to publish the data with a quality statement which indicates non-conformance? The product specification shall describe how each quality element is to be populated, for example, stating the mechanism to reference the quality evaluation procedure, and allowable values for the quality results.

The application schema shall indicate how the data quality elements will be related to the data items, for example whether a particular dataset should have homogeneous quality, or whether quality elements can be related to feature collections, individual feature objects or attributes.

Finally, the encoding description (clause 15) shall indicate how the quality elements will be encoded.

11-9 Data Classification and Encoding Guide

The data product specification shall provide information on how the data is to be captured. This should be as detailed and specific as necessary. The product specification shall include this information for each identified scope.

The product specification includes the collection criteria for mapping real world objects to the conceptual objects of the dataset. Data products can carry information about their data sources (metadata lineage elements); the product specification and application schema will show whether this is expected, and how it is to be done.

Any organization performing data capture for the data product defined by the data product specification shall provide references to any more detailed encoding guide used in addition to that indicated in the product specification for the capturing process.

NOTE A data capture and classification guide is an important part of a data product specification that has to be written before a capturing process can start.

Table 11-8 — Data capture information

Name	Description	Mult	Type
dataSource	Identification of the kinds of data sources usable to product datasets compliant with the considering specification	0..*	CharacterString
productionProcess	Link to a textual description of the production process (including encoding guide) applicable to the datasets compliant with the considering specification	0..*	CharacterString (URL)

11-10 Data Maintenance

The data product specification shall provide information on how the data is maintained. It should describe the principles and criteria applied in maintenance decisions, as well as the expected frequency of updates. The product specification shall include this information for each identified scope.

Maintenance information shall also provide procedures regarding how known errors in the data shall be handled. Any organisation performing data maintenance for the data product defined by the data product specification shall provide a reference to the detailed maintenance guide used for the maintenance process. (See also Metadata / Maintenance Information). Information about maintaining the data product specification itself is included in the Overview.

Table 11-9— Maintenance information

Name	Description	Mult	Type
maintenanceAndUpdateFrequency	Frequency with which changes and additions are made to the data product (per update scope)	1..*	MD_MaintenanceInformation (ISO 19115-1)
dataSource	Identification of the kinds of data sources usable to produce datasets	1..*	LI_Source (ISO 19115-1)
productionProcess	Textual description of the production process applicable to the datasets (per scope or data source)	1..*	LI_ProcessStep (ISO 19115-1)

11-11 Portrayal

The data product specification may provide information on how the data is to be presented as graphic output, e.g. as a plot or as an image. This is an optional section; however it is strongly recommended that it is included where a product specification defines an IHO navigational product. Where included, this shall take the form of a reference to a portrayal library that contains a set of portrayal rules and a set of portrayal specifications (Table 11-10). The product specification shall include this information for each identified scope.

Classes and attributes required to support portrayal for a particular product need to be registered in a feature catalog dictionary and the feature catalogue for that product specification. Examples could be cartographic object classes, scale maximum / minimum attributes, attributes which suggest layout for textual information (for example $TINTS, $JUSTH).

The portrayal library shall be defined in accordance with S-100 Part 9.

Table 11-10— Portrayal Information

Name	Description	Mult	Type
portrayalLibraryCitation	Bibliographic reference to the portrayal library	0..1	CI_Citation (ISO 19115-1)

11-12 Data Product format (encoding)

The data product specification shall define the format (encoding) in which each scope within the data product is delivered.

This section includes a description of file structures and format. The file structure (encoding) could be specified completely here, or by reference to a separate profile or standard. For example, S-100 gives guidance on GML (ISO 19136) encoding; a given product would have a specific GML application schema, expressed in one or more XML Schema Definition Language files. Specialized products may use other encodings, for example S-100 contains a profile of ISO 8211 binary encoding.

Table 11-11 — Data format information

Name	Description	Mult	Type
formatName	Name of the data format	1..*	CharacterString
version	Version of the format (date, number, etc.)	0..1	CharacterString

characterSet	Character coding standard used for the dataset (western European requirement, Greek, Turkish, Cyrillic)	1	MD_CharacterSetCode (ISO 19115-1)
specification	Name of a subset, profile, or product specification of the format	0..1	CharacterString
fileStructure	Structure of delivery file	0..1	CharacterString

11-12.1 Descriptions of GML data formats

Documentation of an encoding based on a GML application schema shall include a description of any constraints required. Examples are:

Whether geometry may be encoded only inline, only by reference, or using either method;

Any constraints on the order of object types in the dataset, for example whether information types must precede spatial and feature data objects;

Schema locations, namespaces, required imports;

Whether validation requires methods in addition to XML schema-based validation, and if so, specifications of the validation rules or a permanent Internet location where they are available for download. For example, rule-based validation may be needed for checking the values of conditional attributes.

11-13 Data product delivery

The data product specification may define the delivery medium for each identified scope. This is an optional section. If a data product can be delivered in different formats then the appropriate information for each shall be given. Data product delivery and medium information are specified in Table 11-12.

Table 11-12 — Delivery Medium Information

Name	Description	Mult	Type
unitsOfDelivery	Description of the units of delivery (for example tiles, geographic areas)	0..1	CharacterString
transferSize	Estimated size of a unit in the specified format, expressed in Mbytes	1	>0
mediumName	Name of the data medium	1	Free text
otherDeliveryInformation	Other information about the delivery	1	Free text

11-14 Additional information

This section of the data product specification is optional and may include any other aspects of the data product not provided elsewhere in this specification. Such aspects may include recommended training, creating or using the product, or details of related products. If this information only applies to a part of the data product, then the scope for this must be clearly identified (Table 11-13).

Table 11-13 — Additional information

Name	Description	Mult	Type
additionalInformation	Any additional information to describe the data product	0..1	CharacterString

11-15 Metadata

The core metadata elements as defined in ISO 19115-1 and S-100 Part 4 (Metadata) shall be included with the data product. Discovery and Quality metadata shall be structured as per S-100 Parts 4a and 4c, respectively. Any additional metadata items required for a particular product specification shall be documented in the data product specification. These should be defined using ISO 19115-1, 19115-2, 19157 (for data quality) and ISO 19115-3, with extensions or restrictions if required. The application schema shall show how metadata is carried in the datasets. This information shall be specified for each identified scope.

11-16 Digital Signatures

Data exchange is implemented using S-100 Part 17. Exchange sets require the use of digital signatures as defined by S-100 Part 15. There is no requirement for Product Specifications to develop or implement other digital signature schemes for data exchange.

Appendix 11-A
Creating an S-100 product specification
(informative)

11-A-1 Introduction

A data product specification is a precise technical description which characterises a geospatial data product. It includes general information for data identification as well as information for data content and structure, reference system, data quality aspects, data capture, maintenance, delivery and metadata.

The process described in this Appendix should be applied to each specification scope identified for the product. For example, if the product will contain a mixture of vector (feature) and coverage data, then the product specification would identify at least two scopes, and the process would be repeated for each scope. If the product contains more than one scope with the same geometry requirement (for example two scopes with vector geometry but different application schemas, or different maintenance regimes), then the process could still be followed twice, taking the same route.

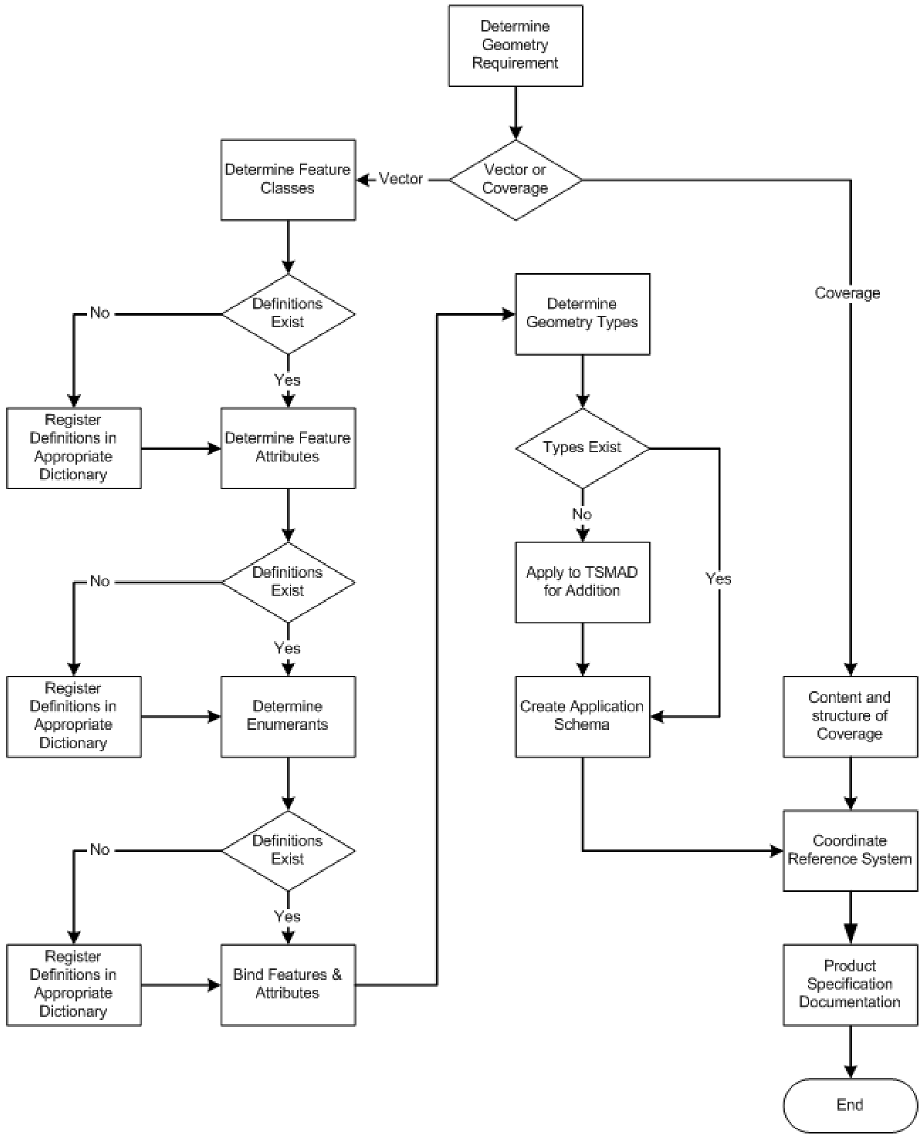


Figure 11-A-1 – Product specification process

The main reason for creating a data product specification is to define the characteristics of a newly developed data product.

11-A-2 General approach

The general approach to creating an S-100 Based product specification is shown in the process flow diagram in Figure 11-A-1. Further information on the processes is given in the following sections.

11-A-2.1 Determine geometry requirement

The first step is to determine whether the scope will be feature based (i.e. use vector geometry) or coverage-based. Certain aspects of a product specification apply only to feature-based data and certain aspects apply only to coverage-based data. A product specification may include both feature-based and coverage-based data, by using specification scopes.

11-A-3 Feature-based product

11-A-3.1 Determine feature attributes

Determine which feature attributes are required in the product. Seek definitions in existing authoritative feature data dictionaries. If required definitions do not exist then define new feature attributes.

11-A-3.2 Determine enumerates

Determine which enumerates are required in the product. Seek definitions in existing authoritative feature data dictionaries. If required definitions do not exist then define new enumerates.

11-A-3.3 Register definitions in appropriate dictionary

If new definitions are required then seek to register them in the most appropriate feature concept dictionary. The IHO will hold one such dictionary. The S-100 Feature Catalogue component does allow for feature or attribute types to be defined locally, if it is not possible to register them in an external dictionary.

11-A-3.4 Bind features and attributes

Features and attributes that are defined in a feature concept dictionary shall be bound in a feature catalogue.

11-A-3.5 Determine geometry types

Determine which geometry types are required in the product. S-100 includes definitions of 1D and 2D geometry types. If a geometry type is required that is not specified in S-100 Part 7 Spatial Component, then apply to S-100WG for it to be added to the framework.

11-A-3.6 Create application schema

It is possible to express an application schema in two different ways:

- Using a conceptual schema language (a logical model);
- Using an encoding specific language (a physical model).

EXAMPLE An example of a conceptual schema language is the UML. An example of an encoding specific language is XML Schema Definition Language.

An S-100 application schema may be expressed using the UML. The resulting model shall be included in the Product Specification so that the logical organisation of the data can be visualised easily. This will be particularly helpful where features have complex structures or relationships. An introduction to UML is included in the S-100 Main Document.

In some cases it is possible to generate the physical application schema automatically from the logical application schema.

EXAMPLE GML is an XML grammar for encoding geographic information. GML application schemas are written using XML Schema Definition Language which is itself a form of XML. Specific rules for designing GML application schemas using UML Class Diagrams are presented in ISO 19136 (the ISO/TC 211 standard for GML). The UML has a standard XML encoding that can be used for interchange of UML models between UML packages. Therefore, if the ISO 19136 rules for designing GML application schemas using UML are adhered to it is possible to export the resulting UML model as XML and to transform the resulting XML to the XML encoding of a GML application schema. The transformation between the UML XML and the GML application schema XML may be undertaken with an XML Stylesheet. Tools have been created that accomplish this task.

Physical encoding mechanisms may define means by which the physical application schema can be used to validate data instances that claim conformance with the application schema in an automatic way.

EXAMPLE GML schemas can be used for a certain amount of dataset validation. The feature and attribute definitions, referenced from the dictionaries, can be presented to the users. GML application schemas are written in XML Schema Definition Language. This is capable of expressing simple constraints, e.g. minimum and maximum values, character patterns. It is not capable of directly expressing constraints which involve more than one property type (for example, "if there is more than one value of 'colour', 'colour pattern' must be set"). If these are included in the Application Schema, perhaps in a formal language such as Object Constraint Language, the ISO 19136 rules ignore them. Thus the GML schema associated with a given product can only be used for a limited validation.

11-A-4 Coverage based product

11-A-4.1 Content and structure of the coverage

The content and structure of a coverage-based product shall be described in terms defined by ISO 19123.

11-A-5 Coordinate Reference System

Determine the appropriate CRS for the data product. More than one CRS may be specified. If necessary, define coordinate operation methods and parameters that shall be used in conjunction with the data product.

Page intentionally left blank

Appendix 11-B
Example Product Specification
(informative)

11-B-1 Overview

11-B-1.1 Product specification metadata

Title		Tide Prediction Information Product Specification
Version		1.0
Date		Created: 2008-01-18
Language		English
Classification		Unclassified
Contact	Organisation Name	Data Product Owner
	Role	Owner
Identifier		IHO:S100:PSExample1
Maintenance		Every five years

11-B-1.2 Product description

Name		Tide Prediction Information
Abstract		Encodes information and parameters for use in making tide predictions
Content		A conformant dataset may contain features associated with the prediction of tides. The specific content is defined by the Feature Catalogue and the Application Schema.
Spatial Extent	Description	Global, marine areas only
	East Bounding Longitude	180
	West Bounding Longitude	-180
	North Bounding Latitude	90
	South Bounding Latitude	-90
Specific Purpose		The data shall be collected for the purpose of tide prediction.

11-B-1.3 Specification scope

This product specification defines only one general scope which applies to all its sections.

Scope Identification	GeneralScope
Level Name	General Scope

11-B-1.4 Data product identification

Title		Tide Prediction Information
Abstract		Encodes information and parameters for use in making tide predictions
Geographic Description	Description	Global, marine areas only
	East Bounding Longitude	180
	West Bounding Longitude	-180
	North Bounding Latitude	90
	South Bounding Latitude	-90
Spatial Resolution	Equivalent Scale	10000
Purpose		The data shall be collected for the purpose of tide prediction.
Language		Not applicable

Data Product Identification Scope: GeneralScope

11-B-2 Data content and structure

11-B-2.1 Data product identification

TPI is a feature-based product. This section contains a feature catalogue and an application schema which is expressed in UML.

11-B-2.2 Feature Catalogue

Name: Tide Prediction Information Feature Catalogue
Scope: Catalogue containing features associated with the prediction of tides.
Field of application: Marine navigation
Version Number: 1.0
Version Date: May 2009
Producer: International Hydrographic Organization
Functional Language: English

Feature Type
Name: Tide Prediction
Definition: Method for calculating tidal motion.
CamelCase: TidePrediction
Remarks: -
Alias: -

Feature Attributes
Name: Object Name
Attribute Type: Simple
Definition: The individual name of an object.
CamelCase: objectName
Cardinality: 0..1

Data Type: text

Name: National Object Name
Attribute Type: Simple
Definition: The individual name of an object in the national Language.
CamelCase: nationalObjectName
Cardinality: 0..1
Data Type: text

Name: Status
Attribute Type: Simple
Definition: The geometric primitive of the associated feature
camelCase: status
Cardinality: 1
Data Type: Enumeration
Values:
1: Permanent
2: Occasional
3: Recommended
4: Not in use
5: Periodic/intermittent
6: Reserved

Name: Method of Tidal Prediction
Attribute Type: Simple
Definition: The technique employed to calculate tidal predictions
camelCase: methodOfTidalPrediction
Cardinality: 1
Data Type: Enumeration
Values:
1: Simplified harmonic
2: Full harmonic
3: Time and height difference

Feature Type
Name: Tide Harmonic Prediction
Definition:
camelCase TideHarmonicPrediction
Remarks: -
Alias: -

Feature Attributes
Name: Value Of Harmonic Constituents
Attribute type: Complex
Definition:
camelCase: valueOfHarmonicConstituents
Cardinality: 1
Data Type: Harmonic Constituent

Name: Harmonic Constituent

Attribute type: Complex
Definition: One of the harmonic elements in a mathematical expression of the tide-producing force, and in corresponding formulae for the tide or tidal stream. Each constituent represents a periodic change of relative position of the Earth, Sun and Moon.
CamelCase: harmonicConstituent
Cardinality: 1..*

Sub Attributes
Name: CategoryOfHarmonicConstituents
Attribute Type: Simple
Data Type: Enumeration
Values:
1: M2
2: S2
3:MM

Name: Constituent Amplitude
Definition: The amplitude of a tidal constituent for a given place in metres
Attribute Type: Simple
Data Type: Real

Name: Constituent Phase
Definition: The phase lag of a tidal constituent at a particular place in degrees
Attribute Type: Simple
Data Type: Real

Feature Type
Name: Tide Non Harmonic Prediction
Definition: Method of tidal prediction made by applying the times of the moon's transits to the mean height of the tide systems of differences to take account of average conditions and various inequalities due to changes in the phase of the moon, declination and parallax of the moon and sun.
camelCase: TideNonHarmonicPrediction
Remarks: -
Alias: -

Name: English Chart Note
Definition: Textual information calling special attention to some fact.
CamelCase: EnglishChartNote
Remarks: -
Alias: -

Name: Reference Station
Definition: Station at which the tidal observations were made.
CamelCase: ReferenceStation
Remarks: -
Alias: -

11-B-2.3 Application Schema

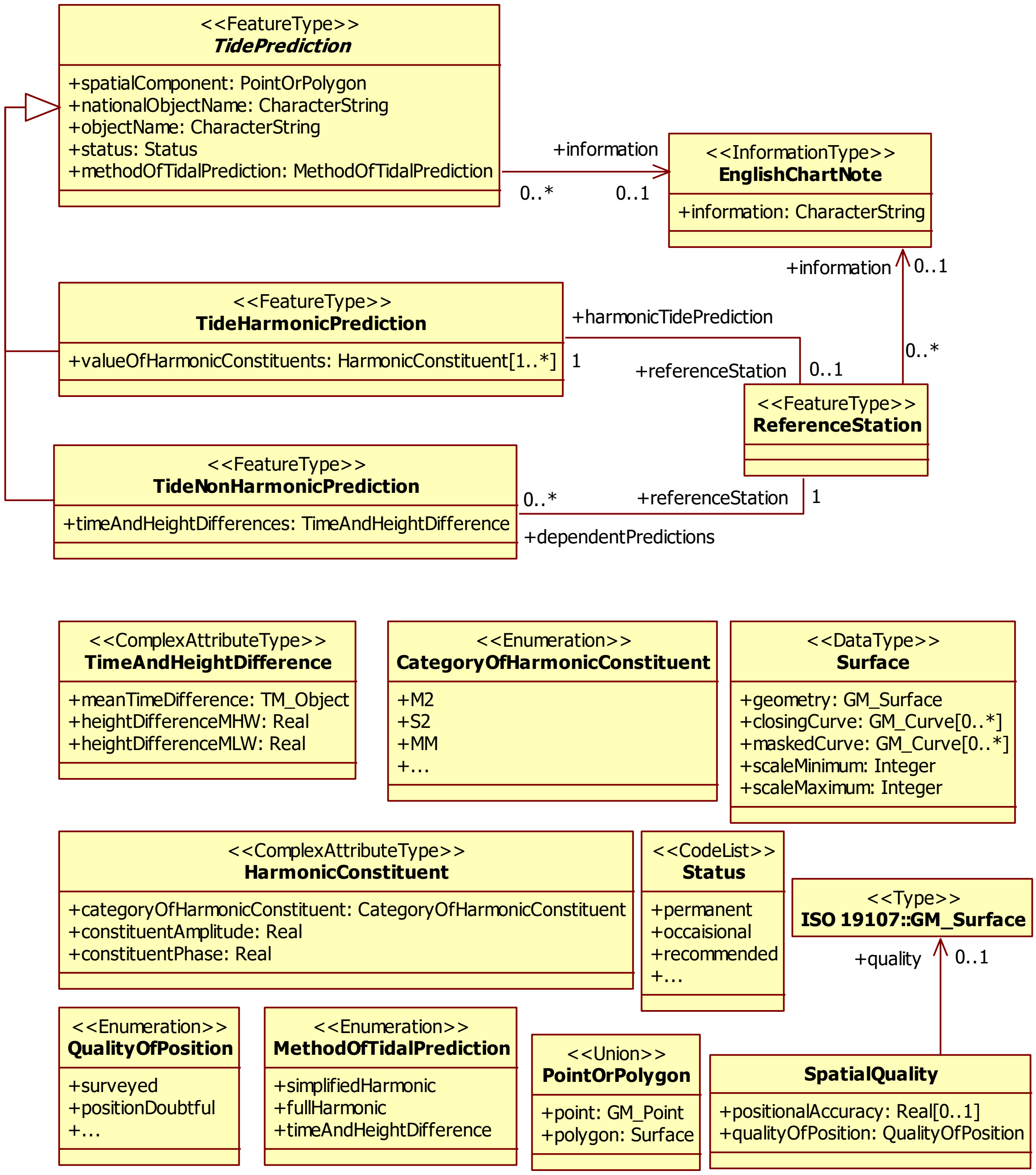


Figure 11-B-1 – Application schema

11-B-3 Data Content and Structure Scope: GeneralScope

11-B-3.1 Coordinate Reference System

Geodetic Coordinate Reference System		
name	code	WGS 84
scope		Horizontal component of the 3D geodetic CRS used by the GPS satellite system.
Geodetic Datum		
scope		Satellite navigation.
Ellipsoid	semiMajorAxis	6378137m
	inverseFlattening	298.257223563
primeMeridian	greenwichLongitude	0°
Ellipsoidal Coordinate System		
Axis 1		
name	code	Geodetic latitude
axisSymbol		Lat
axisDirection		north
unitOfMeasure		angle
Axis 2		
name	code	Geodetic longitude
axisSymbol		Long
axisDirection		east
unitOfMeasure		angle

Coordinate Reference System Scope: GeneralScope

11-B-4 Data Quality

Data Quality Scope: GeneralScope

11-B-5 Data Capture

11-B-5.1 Data source

Tidal predictions are based on a proprietary mathematical model.

11-B-5.2 Production process

A data set conforming to this product specification shall cover an extent of one degree by one degree. Features with surface geometry that cross the edge of product cells shall be split and their geometry shall be specified in the following way, using the class Surface:

Geometry	The polygon geometry specified as the ISO 19107 type GM_Surface
Closing Curve	The segment of the edge of the polygon geometry that coincides with the edge of the cell specified as the ISO 19107 type GM_Curve
Masked Curve	The segment of the edge of the polygon geometry that does not coincide with the edge of the cell specified as the ISO 19107 type GM_Curve

Data Capture Scope: GeneralScope

11-B-6 Data Maintenance

Data are updated as deemed necessary.

Data Maintenance Scope: GeneralScope

11-B-7 Data Product Format

11-B-7.1 Delivery format

Format name	Geography Markup Language
Version	3.1.1
Specification	Geography Markup Language – GML – 3.0, OpenGIS® Implementation Specification, 7 February 2004, OGC Document Number 03-105r1
Language	English
Character Set	utf8

11-B-8 Data Product Delivery

11-B-8.1 Delivery medium

Medium Name	Compact Disc (CD)

Data Product Delivery Scope: GeneralScope

11-B-9 Additional Information

Not applicable.

11-B-10 Metadata

Not Applicable.

Page intentionally left blank

Appendix 11-C
Guidance on Codelists
(informative)

11-C-1 Introduction to Codelists

Product specifications should balance all relevant considerations, for example implementation costs, application operational environment, cross-domain reuse, and reduction of maintenance and distribution efforts, when deciding which approach to use for any particular attribute.

11-C-2 Modelling

When deciding between using a codelist and enumeration, consider the completeness, stability, source, reuse, and application dependencies of the list of values.

- If the set of allowed values is fixed and reasonably short (say, fewer than 20 values?), an enumeration must be used.
- If the list is fixed but long, an enumeration is preferred but a “dictionary model” codelist may be used.
- If only the likely values of an enumeration are known, or the list may be extended by data producers or the user community, a codelist must be used. Whether the “dictionary” or “open” form is preferable depends on who might add values – if it is maintained by an organization, the dictionary form is preferable, if user communities or data producers may add values, the “open” form is preferable.
- If the allowed values change frequently and the list should be updated without major revisions of the product specification, a codelist may be used. The “dictionary” form may be preferable under these circumstances.
- If application logic or portrayal rules depend on values, an enumeration is preferred but a codelist may be used if the logic/rules can be written to cover all possible values (for example, using wildcards or defaults), or otherwise allow graceful recovery from unanticipated values.
- Collections which have internal structure (e.g., types and subtypes of vessels) should be modelled as “dictionary” codelists, pending discussion of the matter by ISO TC211.

11-C-2.1 Hierarchies of codelists

A codelist may also be used as a super-type for more specific codelists. The vocabulary of the super-type is the union of the vocabularies of its sub-types[2]. If additional values are permitted the super-type must be an “open enumeration” or “open dictionary” codelist. Practically, this allows vocabularies developed by different domain expert groups or organizations to be merged.

11-C-3 Codelists maintained by external organizations

If there is an existing well-established codelist maintained by a responsible source, it can be referenced in an application schema. The codelist should meet the following requirements[3]:

- It must be managed by a responsible source – an official national or international standards body, long-established user community, group, or consortium;
- The codelist and its values must be identified by persistent HTTP URIs;
- The list should be well-maintained, meaning all its values must remain available forever, even if they have been deprecated, retired or superseded;
- The list should be in a dictionary language accepted for use in S-10x product specifications.

The IHO may be requested to arrange for the translation, reproduction, and maintenance of codelists meeting only some of the above requirements. Note that this may necessitate a discussion between the IHO and the source.

---

[2] Note that the super-type cannot augment the union set with additional definitions. This conforms to the INSPIRE usage but it may be reconsidered if such augmentation is required at a later time.

[3] Adapted from INSPIRE guidelines.

11-C-4 Data formats of codelist typed attributes

The codelist model in S-100 is designed to be flexible by decoupling application schema from data format to some extent. Data formats may use “codelist extractions” created by extracting codes or values from a codelist dictionary and treat them as ordinary enumerations. The effect is to allow data formats to use either an external dictionary or ordinary enumerations. For example, an XML data format might convert an *ISO3166CountryCodes* codelist maintained by IHO into an XML Schema type:

```
<xs:simpleType name=”ISO3166CountryCodesType”>
  <xs:restriction base=”xs:string”>
    <xs:enumeration value=”EN”/>
    <xs:enumeration value=”FR”/>
    ... other country codes ...
```

As far as implementations using that schema are concerned, it is indistinguishable from an ordinary enumeration. The decision as to which alternative(s) to use in any particular product specification should depend on the circumstances of the data product and its use environment. The decision should be made by the product specification authors when developing the data format. Obviously allowing different data formats to use different representations introduces additional maintenance requirements relating to some data formats, these would be limited to the formats which use “closed” representations (i.e., convert the codelist to an ordinary enumeration).

11-C-4.1 GML and other XML data formats

Enumeration with pattern: The data format in XML schemas must conform to ISO 19136 E.2.4.9, i.e., a union of an enumeration and a pattern of the form:

```
other: [a-zA-Z0-9]+( [a-zA-Z0-9]+)*
```

Examples of use (assuming a codelist which explicitly lists “Norwegian” but not Nynorsk and Bokmål):

```
<language>nor</language>       <!-- Norwegian is an enumerated value -->
<language>other: nno<language> <!-- Norwegian Nynorsk is not an enumerated
value -->
```

External Dictionary: The data format in XML schemas must be the XML Schema built-in types *anyURI*. The use of spaces is discouraged.

Example: (UN/LOCODEs, United Nations Code for Trade and Transport Locations)

In XML schema: Type definition:

```
<xs:simpleType name=”unLoCodeType” type=”xs:anyURI”>
```

and later (in the feature definition):

```
<xs:element name=”unLoCode” type=”unLoCodeType”/>
```

In a dataset:

```
<unLoCode
xlink:href="http://registry.iho.int/codelists/locode/2013/1/USNYC"/>
```

for New York City, identified by code “US NYC” in the UN/LOCodelist version 2013-1 (published July 2013).

11-C-4.2 ISO 8211 encodings

Enumeration with pattern: To accommodate producer-defined values ("other: xyz") this can be encoded either as a "text" type (character string) or as a complex attribute with an integer sub-attribute (for the listed allowed values) and a text sub-attribute (the "other:..." values).

External Dictionary: This can be encoded in two ways:

1. A URI data type with value a URI constructed by combining the URI for the vocabulary (dictionary) and the item code. For example: `http://registry.iho.int/codelists/locode/2013/1/USNYC` for New York City (in the July 2013 edition of UN/LOCODEs list).
2. A complex attribute with two sub-attributes: Vocabulary location (URI) and item code (text). To use the same example: sub-attributes are *vocabulary*= `http://registry.iho.int/codelists/locode/2013/1/` and *itemCode*=USNYC.

The first method is recommended as it reduces data complexity.

11-C-5 Dictionary formats

Use of GML dictionary or SKOS format is recommended. Other formats may be considered under compelling circumstances or after the development of standards in ISO or elsewhere.

11-C-6 Dictionary distribution and discovery

In order to remove dependence on Internet connectivity for interpreting codelist values, codelist dictionaries may be distributed as support files in exchange sets. For the purposes of distribution, discovery, management of updates, and version control, such local dictionary files can be treated as ordinary support files. Discovery metadata for support files is described in Part 4a (see class S100_SupportFileDiscoveryMetadata).

11-C-6.1 Entity resolution with local dictionary files

If mappings from namespaces to dictionary files are needed for a data product, the use of a catalogue file is suggested in which case the product specification may specify the catalogue file name and format. The catalogue file itself can be treated as another support file, having a fixed filename and location in the exchange set which are stated in the product specification.

EXAMPLE A product specification uses XML catalogues for resolving codelist namespaces to local dictionary files. It specifies that the catalogue file shall conform to the OASIS standard for XML catalogues ("XML Catalogs V. 1.1"),

URL: https://www.oasis-open.org/standards#xmlcatalogsv1.1). The product specification standardizes the name of the catalogue file as CODELSTCAT.XML.

Page intentionally left blank

Appendix 11-D
Product Specification Template
(informative)

Introduction

This Appendix is a template for builders of S-100 based Product Specifications. The word version of the template can be downloaded from the following link.

Page intentionally left blank

Appendix 11-E
Guidance on Unique Identifiers
(informative)

A major benefit of the S-100 framework is that products can be produced which can be displayed together on one screen such as in an ECDIS or VTS monitoring system. That necessarily requires a regime which enables an S-100 based system to operate with different products simultaneously. The challenging aspect of operating with different products simultaneously, is to find a solution that allows exactly one instance of a data within the system and which might be simultaneously included in various products. In an S-100 environment, the data originators provide the data and this data could be used in various products without direct influence of a hydrographic office. As long as the data is based on the same framework and if the multiple instances use the same identifier, the data exchange and data processing of this supply chain can be relatively simple.

It is important to preserve original identifiers in data products to assist in identifying data objects which describe the same real-world entity between different datasets, especially datasets from different specifications. For example: Identify instances of the same restricted area between ENC (S-101) and Marine Protected Area (S-122) datasets in an ECDIS. Another principle for preserving instance identifiers is to assist in identifying associated instances between datasets, especially datasets from different specifications. For example: S-124 Navigational Warning marking a light as out of order. This one navigational warning could be used to mark the issue in S-201, S-125 and S-101. Note that this requires the identifiers to be preserved so that the system can link the related feature instances.

Persistent unique Identifiers would reduce the workload and likely issues with translation tables which have to be developed and maintained if various stakeholders use different Identifiers for the same feature; for example, a light has an IALA Identifier (created by a coastal authority) and a HO Identifier. The use of unique Identifiers will likely become more important as interoperability between various products within an S-100 based environment evolves. Thinking interoperability to the last consequence, the clear and standardised definition of the Unique Identifier's structure becomes essential within that structure, and it is recommended that the Maritime Resource Name (MRN) concept, see Part 3 clause 3-10, be utilized as far as possible to have a common system of identifiers within the S-100 regime.

There are implications to establishing a regime of preserving persistent unique identifiers. These include;

- Implications for data maintenance: Processes have to be established to preserve the persistent unique identifiers for features where the identifier is needed, and to do so through maintenance cycles. This means that the identifier remains static throughout the feature lifecycle, even when there are changes to the attributes of the feature. For example, the status of a conspicuous building may change over time, but the building is the same and the identifier should therefore remain static.
- Production processes must be established to preserve the persistent unique identifiers of sources into product instances. If a source object is used to create an amalgamated feature (for example, built up area is made up of all the buildings in the area, but need not show them individually), then the new feature should get a new identifier, and it may not be necessary to preserve the source object identifiers into the product.
- It may be prudent to establish product specific rules for when and how persistent unique identifiers change with object change. For example, a platform is removed; does the remaining obstruction retain the identifier, or is it given a new identifier?
- Persistent unique identifiers may not give any indication of version/date of a feature instance. Guidelines should be established by stakeholders of products and object types for how to determine the most up to date instance if there are discrepancies between data objects which describe the same real-world entity between different datasets.

Persistent unique identifiers are likely to only be unique from the source originator. It is theoretically possible that two source originators generate different feature instances from the same real-world item. It is therefore important that stakeholders communicate, especially among stakeholders that intend to provide data to the same end user systems. Communication should be aimed at understanding domains and working out interoperability issues.

Page intentionally left blank

S-100 – Part 12

S-100 Maintenance Procedures

Page intentionally left blank

Contents

Page intentionally left blank

12-1 Scope

As users begin to implement S-100 and associated Product Specifications, errors and deficiencies in S-100 may be found and these will need to be handled in a uniform manner. This Part specifies procedures to be followed in updating, maintaining and publishing the various parts of S-100. It excludes the maintenance of the IHO Geospatial Information (GI) Registry, as each Register Manager will have their own specific procedures for updating their Register(s). Additionally, this Part excludes the maintenance regime of Product Specifications. However, S-100 versions must be backward compatible to ensure interoperability of Product Specifications.

NOTE: All S-100 based Product Specifications shall include a maintenance section.

12-2 Maintenance Procedures

Change proposals for S-100 are coordinated by S-100WG and shall be made available via the IHO web site. Organizations that wish to make changes to S-100 must address their change proposals to the IHO Secretariat.

Changes to S-100 are classified at one of three different levels: *new edition*, *revision*, or *clarification*. In each case, the development, consultation and approval process will be slightly different, ranging from a very comprehensive regime for *new editions*, to approval at the level of a subordinate body for *clarifications*. *New editions* and *revisions* are considered to be “significant changes” for the purposes of review, consultation and approval.

All proposed changes shall be technically and commercially assessed before approval. All proposals shall be submitted to the secretary S-100WG using the S-100 Maintenance - Change Proposal Form in Appendix 12-A (normative).

Changes to IHO technical standard S-100 shall be subject to the terms of Resolution 2/2007.

12-2.1 Clarification

Clarifications are non-substantive changes to S-100. Typically, *clarifications*: remove ambiguity; correct grammatical and spelling errors; amend or update cross references; insert improved graphics in spelling, punctuation and grammar. A clarification must not cause any substantive semantic change to S-100. *Clarifications* are the responsibility of the relevant subordinate body and may be delegated to the responsible editor.

12-2.2 Revision

Revisions are defined as substantive semantic changes to S-100. Typically, *revisions* change existing specifications to correct factual errors; introduce necessary changes that have become evident as a result of practical experience or changing circumstances; or add new specifications within an existing section. *Revisions* could have an impact on either existing users or future users of a revised standard. It follows that a full consultative process that provides an opportunity for input from as many stakeholders as possible is required. Proposed changes to S-100 should be evaluated and tested wherever practicable. The approval of Member States is required before any *revisions* to S-100 can enter into force. All cumulative *clarifications* must be included with the release of approved corrections revisions.

A *revision* shall not be classified as a *clarification* in order to bypass the appropriate consultation processes.

12-2.3 New Edition

New Editions of S-100 introduce significant changes. *New Editions* enable new concepts, such as the ability to support new functions or applications, or the introduction of new constructs or data types, to be introduced. *New Editions* are likely to have a significant impact on either existing users or future users of the revised standard. It follows that a full consultative process that provides an opportunity for input from as many stakeholders as possible is required. Proposed changes to S-100 should be evaluated and tested wherever practicable. The approval of Member States is required before any *New Edition* of S-100 can enter into force. All cumulative *clarifications* and *revisions* must be included with the release of an approved *New Edition* of S-100.

12-3 Version Control

The IHO shall release new versions of S-100 as necessary. New versions shall include *clarifications*, *revisions* and *new editions*. Each version shall contain a change list that identifies the changes between versions of S-100.

12-3.1 Clarification Version Control

Clarifications shall be denoted as n.n.*n*. Each clarification or set of clarifications approved at a single point in time shall increment *n* by 1.

12-3.2 Revision Version Control

Revisions shall be denoted as n.*n*.0. Each revision or set of revisions approved at a single point in time shall increment *n* by 1. Revision version control shall set clarification version control to 0.

12-3.3 New Edition Version Control

New Editions shall be denoted as *n*.0.0. Each new edition approved at a single point in time shall increment *n* by 1. New Edition version control shall set the clarification and revision version control to 0.

Appendix 12-A
S-100 Maintenance - Change Proposal Form
(normative)

Organization	Date
Contact	Email

Change Proposal Type *Select only one option*

1.Clarification 2.Revision 3.New Edition

Location *Identify all change proposal locations*

S-100 Version No.	Part No.	Section No.	Proposal Summary

Change Proposal

Please provide a detailed change proposal.

Change Proposal Justification

Please provide a suitable explanation for the change and where applicable supporting documentation.

Please send completed forms and supporting documentation to the IHO Secretariat (addt@iho.int).

Page intentionally left blank

S-100 – Part 13

Scripting

Page intentionally left blank

Contents

Page intentionally left blank

13-1 Scope

This Part defines a standard mechanism for including scripting support in S-100 based products. Scripting provides for processing of S-100 based datasets via script files written in the Lua programming language.

13-2 Conformance

Scripts conforming to this part shall be implemented using version 5.1 of the Lua programming language.

13-3 Normative References

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

Lua 5.1 Reference Manual, https://www.lua.org/manual/5.1/

ISO 19125-1:2004, *Geographic information -- Simple feature access -- Part 1: Common architecture*

13-4 Purpose

This Part is provided to permit the normative expression and processing of rules for S-100 based products. Possible usage examples include: portrayal rules, product interoperability rules, rules for detecting navigational hazards, data validation rules, etc.

The use of scripting removes ambiguity from rule expression, ensures consistency among applications, and allows for rules to be modified or extended via catalogue updates.

13-5 Scripting catalogue

A scripting catalogue (see Figure 1) is a collection of script files written for use within a scripting domain.

For instance, portrayal is a scripting domain. The rule files contained within a Lua Portrayal Catalogue comprise a scripting catalogue.

All scripting catalogues are guaranteed to contain the standard catalogue functions defined in clause 13-8.1. Scripting catalogues may additionally contain domain specific catalogue functions. The standard catalogue functions simplify the creation, integration, and testing of scripts within a scripting domain.

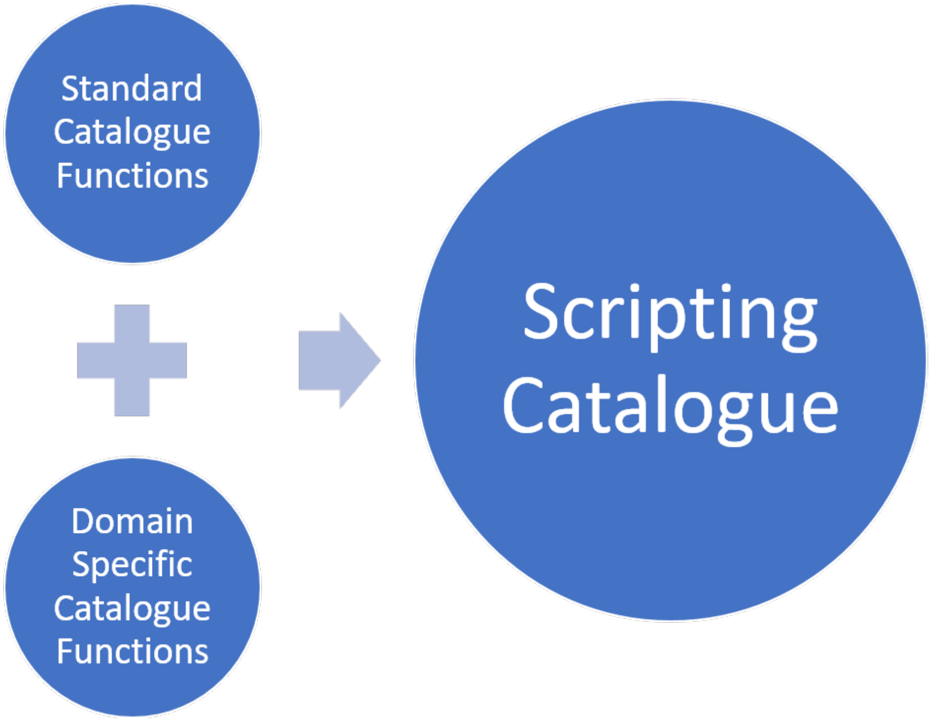


Figure 13-1 – Composition of a Scripting Catalogue

In order to apply rules within a scripting domain, scripting catalogues interact with host functions. The relationship between the scripting catalogue and the host functions is shown in Figure 13-2 below. The host functions serve to decouple the scripting catalogue from the host's implementation of S-100 concepts and functionalities.

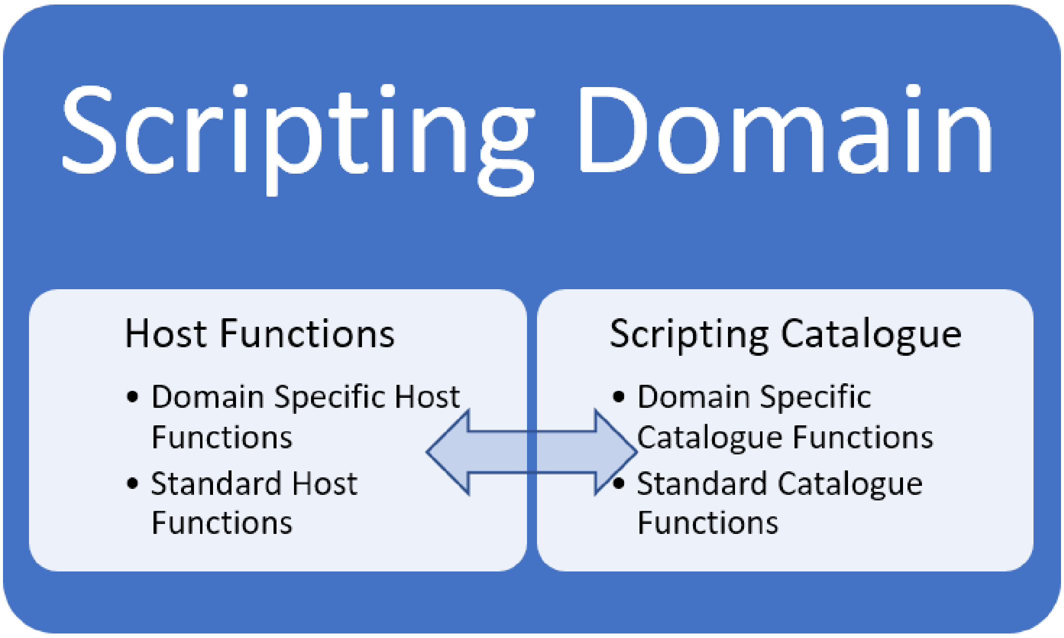


Figure 13-2 - Scripting Catalogue / Host interaction within a Scripting Domain

13-5.1 Distribution

The distribution mechanism of a scripting catalogue is defined within the scripting domain. For example, S-100 Part 9A includes a scripting catalogue within the Portrayal Catalogue; distribution of the scripting catalogue is accomplished via distribution of the Portrayal Catalogue.

Each instance of a scripting catalogue must include all standard catalogue functions.

13-5.2 Domain specific catalogue functions

The standard scripting functions are always available within a scripting catalogue. Parts of S-100 which use scripting may provide additional scripting functions as needed to support domain-specific functionality. In this case, the additional functions are referred to as "domain specific functions".

Domain specific functions intended for host / scripting catalogue interaction (see Figure 2) must be specified within the relevant Part of S-100. Domain specific functions used internally within a scripting catalogue need not be specified within S-100.

For example, assume S-100 Part *N* uses scripting and requires the addition of scripting functions *X*, *Y*, and *Z*. If functions *X* and *Y* are called from the host, but function *Z* is only called from functions *X* and *Y*, S-100 Part *N* must specify required functions *X* and *Y*, and provide the documentation for each function. Since function *Z* is only used internally by the scripting catalogue, it does not need to be documented.

Domain specific functions used for interaction between a host and scripting catalogue are referred to as "domain specific host functions" or "domain specific catalogue functions", depending on whether they are implemented by the host or within the scripting catalogue.

13-6 Data exchange

Data that is passed to the host from a scripting catalogue may be retrieved using the Lua C API functions that correspond to the data type. For the simple data types such as nil, boolean, string and number, retrieval of the data is trivial. For more complex data types, the scripting catalogue encodes the data using the Data Exchange Format (DEF) described in this section.

13-6.1 DEF Schema

The Data Exchange Format (DEF) is a string, formatted as described below. Host parsing of the DEF is simple to implement using the parsing capabilities built into all popular programming languages. Host parsing of the DEF should typically be implemented using string splitting operations such as String.split() in Java, or using simple scan parsing, such as strtok() in C or C++.

A DEF string is a series of one or more elements separated by semicolons (;). Each element is comprised of an item string, followed optionally by a colon (:) and a parameter list. A parameter list is one or more parameter strings separated by commas (,).

Note that string parameters are not surrounded by any delimiters such as quotation marks, however special characters within the string parameters will be escaped using an ampersand (&) as described in clause 13-6.1.2.

13-6.1.1 Special characters

The following Table lists the special characters used by the DEF.

Table 13-1 – Special Characters

Special Character	Usage
Semicolon (;)	Separates the individual elements of a DEF.
Colon (:)	Separates each element into an item string and a parameter list.
Comma (,)	Separates the individual parameters of a parameter list.
Ampersand (&)	Escapes / encodes special characters contained within the DEF.

13-6.1.2 String encoding

Special characters contained within the DEF are escaped / encoded using the character sequences listed in the following Table.

Table 13-2 – String Encoding

Special Character	Encoding
Semicolon (;)	&s
Colon (:)	&c
Comma (,)	&m
Ampersand (&)	&a

For example, a notional DEF containing four elements that might be used to represent drawing instructions:

```
PenWidth:0.64;PenColor:LANDF,0.75;DrawLine;DrawTextStrings:Hello&m
world!,,Foo&cbar
```

The first element has one parameter (0.64), the second element has two parameters (LANDF and 0.75), the third element has no parameters, and the fourth element has three parameters (Hello, world!, null or empty, and Foo:bar).

13-6.1.3 Parsing

There are four steps to parsing the DEF: (1) get each element, (2) get the item and parameters for each element, (3) break the parameters into individual pieces, and then (4) decode each parameter. The notional DEF:

Item1:P1A;Item2:P2A,P2B;Item3:Hello&m world!

The host should first split the DEF into individual elements on each semicolon (;) boundary resulting in the following:

Table 13-3 – Parsing – Step 1

ELEMENT #	ELEMENT
1	*Item1:P1A*
2	*Item2:P2A,P2B*
3	*Item3:Hello&m world!*

Each of the elements should then be divided into an item and the items parameter(s) by splitting on colon (:) boundaries, resulting in:

Table 13-4 – Parsing – Step 2

ELEMENT #	ELEMENT	ITEM	PARAMETERS
1	*Item1:P1A*	*Item1*	*P1A*
2	*Item2:P2A,P2B*	*Item2*	*P2A,P2B*
3	*Item3:Hello&m world!*	*Item3*	*Hello&m world!*

The parameters should then be individually extracted by splitting the parameters on each comma (,) boundary, resulting in:

Table 13-5 – Parsing – Step 3

ELEMENT #	ELEMENT	ITEM	PARAMETER 1	PARAMETER 2	…	PARAMETER N
1	*Item1:P1A*	*Item1*	*P1A*			
2	*Item2:P2A,P2B*	*Item2*	*P2A*	*P2B*		
3	*Item3:Hello&m world!*	*Item3*	*Hello&m world!*			

Once the DEF has been divided into its constituent parts, each parameter should be converted to its original string encoding by performing the substitutions listed in Table 13-2:**Error! Reference source not found.**

Table 13-6 – Parsing – Step 4

ELEMENT #	ELEMENT	ITEM	PARAMETER 1	PARAMETER 2	…	PARAMETER N
1	*Item1:P1A*	*Item1*	*P1A*			
2	*Item2:P2A,P2B*	*Item2*	*P2A*	*P2B*		
3	*Item3:Hello&m world!*	*Item3*	*Hello, world!*			

13-6.2 Attribute path

Scripting catalogues need to be able to determine the value of the attributes on each feature instance contained within a dataset. In order to do so, a catalogue will query the host for each attribute value as needed. When querying a host, the catalogue must identify which attribute of a given feature is being queried. If a feature instance contains only simple attributes, identifying the feature instance and attribute code is sufficient for the host to uniquely identify the requested attribute.

The host requires more information when the attribute value is contained within a complex attribute. For example, consider the following attribute value lookup:

feature.sectorCharacteristic[2].lightSector[1].valueOfNominalRange

Here the feature has a complex attribute *sectorCharacteristic*, which is an array. The second entry of *sectorCharacteristic* contains the complex attribute *lightSector*, the first entry of which contains the simple attribute *valueOfNominalRange*.

When requesting the value of *valueOfNominalRange*, scripting must provide the host with a path to the desired attribute, in addition to the *code* of the desired attribute so that the host can return the actual value. The path is required because the feature instance may have multiple attribute instances with the same *code* contained within alternate attribute paths – for example *feature.simpleAttribute*, vs. *feature.complexAttribute[n].simpleAttribute vs. feature.complexAttribute[n+1].simpleAttribute*.

When the scripting catalogue requests an attribute value from the host, an attribute path is provided to the host using a DEF string. Each section of the path is encoded as an element containing the *AttributeCode* and *Index. AttributeCode* contains the code of a complex attribute; *Index* stores the array index of the complex attribute.

In the example above, the path to *valueOfNominalRange* would be expressed in DEF as follows:

```
sectorCharacteristic:2;lightSector:1
```

The DEF would be used in a call to the host from a scripting catalogue as follows:

```
HostFeatureGetSimpleAttribute(featureID, sectorCharacteristic:2;lightSector:1,
valueOfNominalRange)
```

13-7 Hosting requirements

This section defines the requirements imposed on a host in order to support scripting functionality. For example, a program written to display an S-101 ENC using the S-100 Part 9A portrayal must conform to the requirements of this section.

13-7.1 Lua version

The host must provide a scripting engine; a Lua version 5.1 interpreter or virtual machine. The reference implementation is available from lua.org (http://www.lua.org/). Embedding the reference implementation into the host is recommended. For maximum performance the host can embed or implement a Lua compiler such as LuaJIT (http://luajit.org/).

Further guidance on embedding is provided in *Programming in Lua – Part IV (The C API)*, details of which are available at https://www.lua.org/pil/.

13-7.2 Character encoding

All strings exchanged between the host and the scripting catalogue must be UTF-8 encoded.

13-7.3 Error handling

When calling Lua scripting catalogue functions from the host, a return value of **LUA_OK** from *lua_pcall* indicates success. Otherwise, the standard Lua error handling mechanism is used. An error code is returned to the host and a string detailing the error will be available on the top of the stack.

13-7.4 Array parameters

Several of the scripting catalogue functions expect arrays to be passed as parameters. The arrays are standard Lua arrays which should be created using the Lua C API array functions as documented in *Programming in Lua – Part IV (The C API).*

13-7.5 Host functions

The host must provide the standard host functions detailed in clause 13-8.2.

The host must also provide domain specific host functions in order to support domain specific functionalities. Domain specific functionalities which are unused by the host do not need to be provided. Documentation for domain specific host functions is provided in the Part(s) of S-100 describing the domain specific functionality.

13-7.5.1 Compatibility

The host must guarantee backwards compatibility of the host provided functions with all previously published scripting catalogues. That is, when implementing function *X*, the host must only call scripting catalogue functions which were available in the version of S-100 when *X* was added.

Failure to conform to this requirement may result in incompatibilities when the host attempts to run older scripting catalogues.

13-7.5.1.1 Scripting catalogue / host incompatibility

As new versions of S-100 are published, scripting functions may be added. Scripting functions will never be removed from S-100, although the use of a particular function may be deprecated.

Although backwards compatibility is guaranteed, newer scripting catalogues may attempt to call host functions which are unsupported by the current host. This situation is indicative of a host which has not been updated with the latest host scripting functions. To limit the occurrence of such cases, scripting catalogues should be written using the earliest subset of scripting functions possible.

Scripting incompatibilities (missing host functions) are indicated during scripting initialization. Incompatibility is indicated to the host by returning **LUA_ERRERR** from *lua_pcall*; the error string at the top of the stack will detail the cause of the incompatibility. When this occurs the host should revert to an earlier version of the scripting catalogue if available. It is also recommended to alert the user to check for an update of the host software.

13-8 Standard script functions

This section describes the set of standard script functions which constitute the scripting system. There are two sets of functions described: standard catalogue functions, and standard host functions. The realization of a scripting catalogue only exists within a scripting domain.

Standard catalogue functions, as described in clause 13-8.1, are provided within each scripting catalogue. Standard host functions, as described in clause 13-8.2, are to be implemented by the program which is hosting the scripting environment.

Figure 13-3 below shows the location of each type of scripting function within the scripting environment.

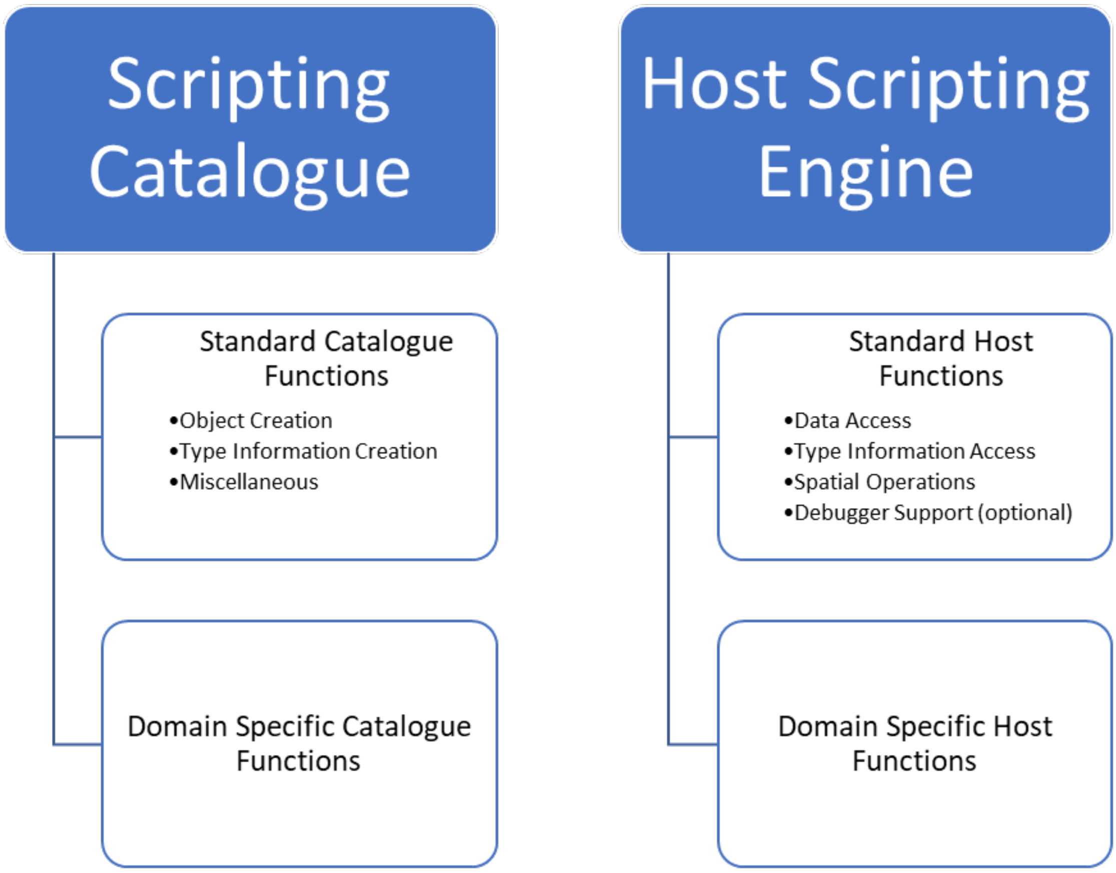


Figure 13-3 – Location of script functions within the scripting environment

Each standard script function is described below on its own sub-clause. A description of the functions purpose, along with a description of the parameters and return value are provided. For clarity, *void* is used to indicate that a function has no return value.

Function parameters which can accept multiple types will be indicated as *variant*. *variant* will also be used if the function can return more than one type. For instance, a function which accepts both integers and strings for its first parameter, and returns either an integer or string dependent on the type passed for the first parameter, would have a signature of:

variant **Function**(variant *param1*)

The function description will indicate the types which are permitted for the *variant* parameter(s).

Many of the standard script functions accept a *featureID, informationTypeID* or *spatialID* parameter. The host must ensure that these various *ID* parameters uniquely identify a single instance among all datasets across all product types to be used by the host during a scripting session. Since each type of *ID* is a string, one way to accomplish this is by prepending the relevant information to the *ID*; for example, "S101.101US003DE01M__.000.F1" to identify the first feature in the referenced S-101 dataset.

13-8.1 Standard catalogue functions

This section describes the standard set of functions which are provided by all scripting catalogues.

All strings passed to these functions must be UTF-8 encoded.

When calling these functions, attribute values are always passed from the host to the scripting environment using strings. This allows values which don't have Lua equivalents to be passed unambiguously. This also allows for decimal values to be passed without the loss of precision which can occur during conversion to IEEE floating point types.

If an attribute value is present but unknown, the value returned from *GetUnknownAttributeString()* should be used.

The following Table shows the string representations of the Types defined by *S100_CD_AttributeValueType*.

Table 13-7 – String representation of types defined by *S100_CD_AttributeValueType*

S100_CD_AttributeValueType	Representation
boolean	"0" represents False "1" represents True
enumeration	S100_FC_ListedValue:code. Do not use S100_FC_ListValue:label
integer	String representation of a signed integer
real	String representation of a decimal number. Trailing zeros are permitted only if significant
text	As provided
date	Character encoding shall follow the format for date as specified by ISO 8601
time	Character encoding shall follow the format for time as specified by ISO 8601
dateTime	Character encoding shall follow the format for date and time as specified by ISO 8601
URI	Character encoding shall follow the format for URI as specified by RFC 3986
URL	Character encoding shall follow the format for URL as specified by RFC 3986
URN	Character encoding shall follow the format for URN as defined by RFC 2141
S100_CodeList	As provided
S100_TruncatedDate	As provided

13-8.1.1 Object creation functions

These functions relieve the host from the burden of constructing Lua tables corresponding to complex types used within the scripting catalogue. They allow the host to create objects which will be passed into the scripting catalogue. The schema and contents of the created objects are opaque to the host – they are only intended for use within the scripting catalogue.

13-8.1.1.1 AttributeConstraints CreateAttributeConstraints(integer *stringLength*, string *textPattern*, string *rangeLower*, string *rangeUpper*, string *rangeClosure*, integer *precision*)

Return Value

AttributeConstraints

A Lua table containing an attribute constraints object.

Parameters

stringLength: integer or nil

The maximum number of characters that may be assigned to the text attribute type. If this value is nil, the length is unconstrained.

textPattern: string or nil

A regular expression defining the structure of text values that may be assigned to the attribute. If this value is nil, the structure is unconstrained.

W3C XML Standard Part 2 Appendix F (Regular Expressions) shall be used to define the text pattern.

rangeLower: string or nil

Specifies the lower range of allowed values for the attribute. If this value is nil, there is no lower value constraint.

rangeUpper: string or nil

Specifies the upper range of allowed values for the attribute. If this value is nil, there is no upper value constraint.

rangeClosure: string or nil

Defines the closure operations for the lower and upper ranges. This is one of enumerated values as defined in Table 1-3. This must be specified if either or both the lower or upper ranges are specified.

precision: integer or nil

If specified, indicates the precision of a real number.

Remarks

Called from the host to create attribute constraints for use by the scripting catalogue.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.2 SpatialAssociation CreateSpatialAssociation(string *spatialType*, string *spatialID*, string *orientation*, variant *scaleMinimum*, variant *scaleMaximum*)

Return Value:

SpatialAssociation

A Lua table containing a spatial association object.

Parameters:

spatialType: string

The type of the spatial. One of: "Point", "MultiPoint", "Curve", "CompositeCurve", or "Surface".

spatialID: string

Used by the host to uniquely identify a spatial.

orientation: string

Orientation of the spatial. One of Forward or Reverse.

scaleMinimum: integer or nil

Minimum display scale for the spatial or nil.

scaleMaximum: integer or nil

Maximum display scale for the spatial or nil.

Remarks:

Called from the host to create a spatial association for use by the scripting catalogue.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.3 Point CreatePoint(string *x*, string *y*, variant *z*)

Return Value:

Point

A Lua table containing a point object.

Parameters:

x: string

X coordinate for the point.

Y: string

Y coordinate for the point.

Z: string or nil

Z coordinate for the point. For 2D points, this value shall be *nil*.

Remarks:

x, y and *z* are expressed using the *real* string representation as described in clause 13-8.1

Called from the host to create a point spatial object for use by the scripting catalogue.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.4 MultiPoint CreateMultiPoint(Point[] *points*)

Return Value:

MultiPoint

A Lua table containing a multipoint object.

Parameters:

points: Point[]

A Lua array of points. The host creates each point by calling *CreatePoint*.

Remarks:

Called from the host to create a multipoint spatial object for use by the scripting catalogue.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.5 CurveSegment CreateCurveSegment(Point[] *controlPoints*, string *interpolation*)

Return Value:

CurveSegment

A Lua table containing a curve segment object.

Parameters:

controlPoints: Point[]

Array of points that define the control points of the curve segment. The host creates each controlPoint by calling *CreatePoint*.

Interpolation: string

The interpolation to use when connecting the control points. One of S100_CurveInterpolationL:name.

Remarks:

Called from the host to create a curve segment spatial object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.6 Curve CreateCurve(SpatialAssociation *startPoint*, SpatialAssociation *endPoint*, CurveSegment[] *segments*)

Return Value:

Curve

A Lua table containing a curve object.

Parameters:

startPoint: SpatialAssociation

Start point for the curve. Host creates by calling *CreateSpatialAssociation*.

endpoint: SpatialAssociation

End point for the curve. Host creates by calling *CreateSpatialAssociation*.

segments: CurveSegment[]

An array of curve segments comprising the curve. Each array entry is created by calling *CreateCurveSegment*.

Remarks

Called from the host to create a curve spatial object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.7 CompositeCurve CreateCompositeCurve(SpatialAssociation[] *curveAssociations*)

Return Value:

CompositeCurve

A Lua table containing a composite curve object.

Parameters:

curveAssociations: SpatialAssociation[]

Array of spatial associations that define the elements of the composite curve. The host creates each SpatialAssociation by calling *CreateSpatialAssociation*.

Remarks:

Called from the host to create a composite curve spatial object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.8 Surface CreateSurface(SpatialAssociation *exteriorRing*, variant *interiorRings*)

Return Value:

Surface

A Lua table containing a surface object.

Parameters:

exteriorRing: SpatialAssociation

The spatial association of the ring that defines the exterior ring of the surface. Host creates by calling *CreateSpatialAssociation*.

interiorRings: SpatialAssociation[] or nil

Defines the “holes” within the surface. Host creates each interior ring by calling *CreateSpatialAssociation*. If there are no holes, this parameter is nil.

Remarks:

Called from the host to create a surface spatial object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.9 ArcByCenterPoint CreateArcByCenterPoint(SpatialAssociation *centerPoint*, real *radius,* real *startAngle,* real *angularDistance*)

Return Value:

ArcByCenterPoint

A Lua table containing an ArcByCenterPoint object.

Parameters:

centerPoint: SpatialAssociation

The spatial association of the point that defines the centre point of the arc. Host creates by calling *CreateSpatialAssociation*.

radius: real

Defines the geodesic distance from the centre.

startAngle: real

Starting bearing of the arc in degrees, range limited to [0.0, 360.0].

angularDistance: real

Angular distance of the arc in degrees, range limited to [-360.0, 360.0]. Positive numbers indicate a clockwise direction.

Remarks:

Called from the host to create an ArcByCenterPoint spatial object. The arc starts at the bearing given by the *startAngle* parameter and ends at the bearing calculated by adding the value of the *angularDistance* parameter to the start angle. The direction of the arc is given by the sign of the angular distance. Bearings are relative to true north except that arcs centred at either pole (where true north is undefined or ambiguous) shall use the prime meridian as the reference direction.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.10 CircleByCenterPoint CreateCircleByCenterPoint(SpatialAssociation *centerPoint*, real *radius*, real *startAngle,* real *angularDistance*)

Return Value:

CircleByCenterPoint

A Lua table containing a CircleByCenterPoint object.

Parameters:

centerPoint: SpatialAssociation

The spatial association of the point that defines the centre point of the circle. Host creates by calling *CreateSpatialAssociation*.

radius: real

Defines the geodesic distance from the centre.

startAngle: real

Optional. Starting bearing of the arc in degrees, range limited to [0.0, 360.0]. Default is zero.

angularDistance: real

Optional. Angular distance of the circle in degrees, must be either -360.0 (counter-clockwise) or 360.0 (clockwise). Positive numbers indicate a clockwise direction. Default is 360 (clockwise).

Remarks:

Called from the host to create a CircleByCenterPoint object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.11 SplineCurve CreateSplineCurve(Point[] *controlPoints*, string *interpolation,* integer *degree,* Knot[] *knots*, KnotType *knotSpec*, boolean *isRational*)

Return Value:

SplineCurve

A Lua table containing a spline curve.

Parameters:

controlPoints: Point[]

Array of points that define the control points of the curve segment. The host creates each controlPoint by calling *CreatePoint*. The number of control points must be three or greater.

interpolation: string

The interpolation to use when connecting the control points. One of S100_CurveInterpolation:name.

degree: integer

The degree of the polynomials used for defining the interpolation.

knots: Knot[]

Array of knots. Each knot defines a parameter in the parameter space of the spline that is used to define the spline basis function. Each knot is created by calling *CreateKnot.*

knotSpec: KnotType

Type of knot distribution in defining the spline. Defined by S100_GM_KnotType.

isRational: boolean

Indicates whether the spline uses rational functions to define the curve.

Remarks:

Called from the host to create a spline curve spatial object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.12 PolynomialSpline CreatePolynomialSpline(Point[] *controlPoints*, string *interpolation,* integer *degree,* Knot[] *knots*, KnotType *knotSpec*, Vector[] *derivativeAtStart*, Vector[] *derivativeAtEnd*, integer *numDerivativeInterior*)

Return Value:

PolynomialSpline

A Lua table containing a polynomial spline.

Parameters:

controlPoints: Point[]

Array of points that define the control points of the curve segment. The host creates each controlPoint by calling *CreatePoint*. The number of control points must be three or greater.

interpolation: string

The interpolation to use when connecting the control points. One of S100_CurveInterpolation:name.

degree: integer

The degree of the polynomials used for defining the interpolation.

knots: Knot[]

Array of knots. Each knot defines a parameter in the parameter space of the spline that is used to define the spline basis function. Each knot is created by calling *CreateKnot.*

knotSpec: KnotType

Type of knot distribution in defining the spline. Defined by S100_GM_KnotType.

derivativeAtStart: Vector[]

Array of Vector that defines the values used for the initial derivative used for interpolation in this curve at the start point of the spline. Up to *degree* - 2 vectors can be defined. Each vector is created by calling *CreateVector.*

derivativeAtEnd: Vector[]

Array of Vector that defines the values used for the final derivative used for interpolation in this curve at the end point of the spline. Up to *degree* - 2 vectors can be defined. Each vector is created by calling *CreateVector.*

numDerivativeInterior: KnotType

The number of continuous derivatives required at interior knots.

Remarks:

Called from the host to create a polynomial spline spatial object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.13 Knot CreateKnot(string *value*[, integer *multiplicity*])

Return Value:

Knot

A Lua table containing a knot object.

Parameters:

value: string

Value of the knot.

multiplicity: integer

The multiplicity of the knot. If omitted, the multiplicity is one.

Remarks:

value is expressed using the real string representation as described in clause 13-8.1.

Called from the host to create a knot object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.1.14 Vector CreateVector(Point *origin,* string[] *offset*, integer *dimension,* string *coordinateSystem*)

Return Value:

Knot

A Lua table containing a knot object.

Parameters:

origin: Point

The location of the point on the GeometricReferenceSurface for which the vector is a tangent.

offset: string[]

Local tangent vector in terms of the differentials of the local coordinates. The offset values are the magnitude of the vector along each coordinate axis.

dimension: integer

The dimension of the origin.

coordinateSystem: string

The coordinate system of the origin (e.g. EPSG:4326).

Remarks:

offset values are expressed using the real string representation as described in clause 13-8.1.

Called from the host to create a vector object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2 Type information creation functions

These functions relieve the host from the burden of constructing Lua tables corresponding to complex types used within the scripting catalogue. They allow the host to create objects used when calling into the scripting catalogue. The schema and contents of the created objects are opaque to the host – they are only intended for use within the scripting catalogue.

The complex types correspond to the classes described in S-100 Part 5 - *Feature Catalogue*. Each type information creation function described in this section specifies the corresponding S-100 Part 5 Feature Catalogue type.

Creation functions for *FC_DefinitionReference* and its dependent types, including the *CI_Citation* class, are intentionally omitted. There are no identified use cases for *FC_DefinitionReference*, and the implementation of *CI_Citation* would be more complicated than the entirety of this section as currently defined.

13-8.1.2.1 Item CreateItem(string *code,* string *name*, string *definition,* string *remarks,* string[] *alias*)

Return Value:

Item

A Lua table containing an item corresponding to an *S100_FC_Item*.

Parameters:

code: string

Code that uniquely identifies the named type within the Feature Catalogue.

name: string

Name of the item.

definition: string

Definition of the named type in a natural language.

remarks: string

Optional. Further explanation about the item.

alias: string[]

Equivalent name(s) of this item.

Remarks:

Called from the host to create an item.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.2 NamedType CreateNamedType(Item *item,* boolean *abstract*, AttributeBinding[] *attributeBindings*)

Return Value:

NamedType

A Lua table containing a named type corresponding to an *S100_FC_NamedType*.

Parameters:

item: Item

Instance of an item created by calling *CreateItem()*.

abstract: boolean

Indicates if instances of this named type can exist in a geographic data set. Abstract types cannot be instantiated but serve as base classes for other (non-abstract) types.

attributeBindings: AttributeBinding[]

An array of zero or more bindings to attributes which describe the characteristic of this named type.

Remarks:

Called from the host to create a named type.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.3 ObjectType CreateObjectType(NamedType *namedType,* InformationBinding[] *informationBindings*)

Return Value:

ObjectType

A Lua table containing an object type corresponding to an *S100_FC_ObjectType*.

Parameters:

namedType: NamedType

Instance of a named type created by calling *CreateNamedType()*.

informationBindings: InformationBinding[]

An array of zero or more bindings to information types that can be associated to this object type by means of an information association.

Remarks:

Called from the host to create an object type.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.4 InformationType CreateInformationType(ObjectType *objectType,* string *superType*, string[] *subType*)

Return Value:

InformationType

A Lua table containing an information type corresponding to an S100_FC_InformationType.

Parameters:

objectType: ObjectType

Instance of a named type created by calling *CreateObjectType()*.

superType: string

Optional. Indicates the code of the information type from which this type is derived.

subtype: string[]

An array of zero or more information type codes which are derived from this type.

Remarks:

Called from the host to create an information type.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.5 FeatureType CreateFeatureType(ObjectType *objectType,* string *featureUseType*, string[] *permittedPrimitives*, FeatureBinding[] *featureBindings*, string *superType*, string[] *subType*)

Return Value:

FeatureType

A Lua table containing a feature type corresponding to an *S100_FC_FeatureType*.

Parameters:

objectType: ObjectType

Instance of a named type created by calling *CreateObjectType()*.

featureUseType: string

A *S100_CD_FeatureUseType:Name*.

permittedPrimitives: string[]

An array specifying zero or more allowed spatial primitive types for the feature type. Each entry is a *S100_FC_SpatialPrimitiveType:Name*.

featureBindings: FeatureBinding[]

An array of zero or more bindings to feature types that can be related to this feature type by means of a feature association.

superType: string

Optional. Indicates the code of the feature type from which this type is derived. The sub-type will inherit all properties from its super-type: Name, definition and code will usually be overridden by the sub-type, although new properties may be added to the sub-type.

subType: string[]

An array of zero or more feature type codes which are derived from this type.

Remarks:

Called from the host to create a feature type.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.6 InformationAssociation CreateInformationAssociation(NamedType *namedType,* Role[] *roles*, string *superType*, string[] *subType*)

Return Value:

InformationAssociation

A Lua table containing an information association corresponding to an S100_FC_InformationAssociation.

Parameters:

namedType: NamedType

Instance of a named type created by calling *CreateNamedType()*.

roles: Role[]

An array of zero to two roles of the association.

superType: string

Optional. Indicates the code of the information association from which this association is derived.

subType: string[]

An array of zero or more information association codes which are derived from this association.

Remarks:

Called from the host to create an information association.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.7 FeatureAssociation CreateFeatureAssociation(NamedType *namedType,* Role[] *roles*, string *superType*, string[] *subType*)

Return Value:

FeatureAssociation

A Lua table containing a feature association corresponding to an S100_FC_FeatureAssociation.

Parameters:

namedType: NamedType

Instance of a named type created by calling *CreateNamedType()*.

roles: Role[]

An array of zero to two roles of the association.

superType: string

Optional. Indicates the code of the feature association from which this association is derived.

subType: string[]

An array of zero or more feature association codes which are derived from this association.

Remarks:

Called from the host to create a feature association.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.8 Role CreateRole(Item *item*)

Return Value:

Role

A Lua table containing a role corresponding to a *S100_FC_Role*.

Parameters:

item: Item

Instance of an item created by calling *CreateItem()*.

Remarks:

Called from the host to create a role.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.9 SimpleAttribute CreateSimpleAttribute(Item *item*, string *valueType,* string *uom*, string *quantitySpecification,* AttributeConstraints *attributeContraints,* ListedValue[] *listedValues*)

Return Value:

SimpleAttribute

A Lua table containing a simple attribute corresponding to a *S100_FC_SimpleAttribute*.

Parameters:

item: string

Instance of an item created by calling *CreateItem()*.

valueType: string

The value type of this feature attribute. A *S100_CD_AttributeValueType:Name*.

uom: string

Optional. Unit of measure used for values of this feature attribute. A *S100_UnitOfMeasure:Name*.

quantitySpecification: string

Optional. Specification of the quantity. A *S100_CD_QuantitySpecification:Name*.

attributeContraints: AttributeConstraints

Optional. Constraints which may apply to the attribute. Created by calling *CreateAttributeConstraints()*.

listedValues: ListedValue[]

Array of zero or more listed values for an enumerated attribute domain. Each listed value is created by calling *CreateListedValue()*. Applies only if *valueType* is *Enumeration* or *S100_Codelist* (with *codelistType* of open enumeration).

Remarks:

Called from the host to create a simple attribute type info object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.10 ComplexAttribute CreateComplexAttribute(Item *item*, AttributeBinding[] *subAttributeBindings*)

Return Value:

ComplexAttribute

A Lua table containing a complex attribute corresponding to a *S100_FC_ComplexAttribute*.

Parameters:

item: Item

Instance of an item created by calling *CreateItem()*.

subAttributeBindings: AttributeBinding[]

An array of one or more of attribute bindings to the sub-attributes.

Remarks:

Called from the host to create a complex attribute type info object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.11 ListedValue CreateListedValue(string *label,* string *definition*, integer *code,* string *remarks,* string[] *aliases*)

Return Value:

ListedValue

A Lua table containing a listed value corresponding to an *S100_FC_ListedValue*.

Parameters:

label: string

Descriptive label that uniquely identifies one value of the feature attribute.

definition: string

Definition of the listed value in a natural language.

code: integer

Numeric code that uniquely identifies the listed value for the corresponding feature attribute. Positive integer.

remarks: string

Optional. Further explanation about the listed value.

aliases: string[]

Optional. Array of zero or more equivalent name(s) of this listed value.

Remarks:

Called from the host to create a listed value type info object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.12 AttributeBinding CreateAttributeBinding(string *attributeCode*, integer *lowerMultiplicity,* integer *upperMultiplicity*, boolean *sequential*, integer[] *permittedValues*)

Return Value:

AttributeBinding

A Lua table containing an attribute binding corresponding to an *S100_FC_AttributeBinding*.

Parameters:

attributeCode: string

The code of the complex or simple attribute that is bound to the item or complex attribute.

lowerMultiplicity: integer

The minimum number of required occurrences of this attribute. This is zero for optional attributes.

upperMultiplicity: integer

The maximum number of allowed occurrences of this attribute. This is nil for an infinite number of allowed attributes.

sequential: boolean

Describes if the sequence of the attributes is meaningful or not. Applies only to attributes which may occur more than once.

permittedValues: integer[]

Array of zero or more permissible values of the attribute. Each entry is a *S100_FC_ListedValue:code*. Applies only to attributes of data type enumeration.

Remarks:

Called from the host to create an attribute binding object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.13 InformationBinding CreateInformationBinding(string[] *informationTypeCode*, integer *lowerMultiplicity,* integer *upperMultiplicity*, string *roleType*, string *role*, string *association*)

Return Value:

InformationBinding

A Lua table containing an information binding corresponding to a *S100_FC_InformationBinding*.

Parameters:

informationTypeCode: string[]

An array containing one or more *S100_FC_InformationType:code* indicating the target information type(s).

lowerMultiplicity: integer

The minimum number of required occurrences of this attribute. This is zero for optional attributes.

upperMultiplicity: integer

The maximum number of allowed occurrences of this attribute. This is nil for an infinite number of allowed attributes.

roleType: string

The nature of the association end. A *S100_FC_RoleType:Name*.

role: string

Optional. The code of the role used for the binding. It must be part of the association used for the binding and defines the end of the association.

association: string

The code of the information association used for the binding; defining also the role.

Remarks:

Called from the host to create an information binding object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.2.14 FeatureBinding CreateFeatureBinding(string[] *featureTypeCode*, integer *lowerMultiplicity,* integer *upperMultiplicity*, string *roleType*, string *role*, string *association*)

Return Value:

FeatureBinding

A Lua table containing a feature binding corresponding to a *S100_FC_FeatureBinding*.

Parameters:

featureTypeCode: string[]

An array containing one or more S100*_FC_FeatureType:code* indicating the target feature type(s).

lowerMultiplicity: integer

The minimum number of required occurrences of this attribute. This is zero for optional attributes.

upperMultiplicity: integer

The maximum number of allowed occurrences of this attribute. This is nil for an infinite number of allowed attributes.

roleType: string

The nature of the association end. A *S100_FC_RoleType:Name*.

role: string

The code of the role used for the binding. It must be part of the association used for the binding and defines the end of the association.

association: string

The code of the feature association used for the binding.

Remarks:

Called from the host to create a feature binding object.

It is not intended that the host manipulate the returned object; the object is intended to be passed from the host back to the scripting catalogue.

13-8.1.3 Miscellaneous functions

The functions described on the following pages do not fall under one of the previously described functionalities.

13-8.1.3.1 string GetUnknownAttributeString()

Return Value:

string

A string that represents an attribute value that is present but unknown.

Remarks:

Intended to permit differentiation of unknown string values from empty string values. This function returns a constant value.

13-8.1.3.2 string EncodeDEFString(string *input*)

Return Value:

string

An encoding of *input* as described in clause 0.

Parameters

input: string

The unencoded string.

Remarks:

Encodes the input string as described in section 0.

13-8.1.3.3 string DecodeDEFString(string *encodedString*)

Return Value:

string

A decoded version of *encodedString*.

Parameters

encodedString: string

The encoded string.

Remarks:

Decodes an input string which was previously encoded as described in section 0.

13-8.1.3.4 void TypeSystemChecks(boolean *enabled*)

Return Value:

None

Parameters

enabled: boolean

Enables or disables type checking.

Remarks:

Indicates the data type of each parameter should be verified on each function call. Disabled by default. Type system checks may be enabled during catalogue development as a debugging aid.

13-8.2 Standard host functions

The host must provide a set of "callback" functions that provide the scripting environment with: Access to the host's realization of the S-100 General Feature Model; access to type information for any entity defined by the model; and access to spatial operations which can be used to perform relational tests and operations on spatial elements defined by the model. The host may optionally provide a callback function used to interact with a debugger.

Offloading these tasks to the host, rather than providing rigid data structures which are passed between the host and scripting, allows the host to interact with scripting using the hosts optimal representation of the General Feature Model. Host translation of its internal data model to a particular input schema is not necessary when using scripting.

Any of the standard host functions may be called from the scripting catalogue during the execution of a script.

13-8.2.1 Data access functions

The host must implement the functions described on the following pages to allow the scripting environment to access data the host has loaded from a dataset. These functions provide the scripting environment with access to features, spatials, attribute values, and information associations.

13-8.2.1.1 string[] HostGetFeatureIDs()

Return Value:

string[]

A Lua array containing all of the feature IDs in the dataset.

Remarks:

Instructs the host to return all feature IDs relevant to the current scripting catalogue operation. This would typically be all of the features in an *S100_Dataset* or *S100_DataCoverage*.

As discussed in clause 13-8, the host is responsible for ensuring each feature ID uniquely identifies a single feature instance among all product types and datasets to be used during the current scripting session.

13-8.2.1.2 string HostFeatureGetCode(string *featureID*)

Return Value:

string

The code defined by the Feature Catalogue for the feature type of the feature instance.

Parameters:

featureID: string

Used by the host to uniquely identify a feature instance.

Remarks:

Instructs the host to return the feature type code for the feature instance identified by *featureID*.

13-8.2.1.3 string[] HostGetInformationTypeIDs()

Return Value:

string[]

A Lua array containing all of the information type IDs in the dataset.

Remarks:

Allows scripts to query the host for a list of information types contained within a given dataset.

Instructs the host to return an array containing all information IDs in the given dataset.

13-8.2.1.4 string HostInformationTypeGetCode(string *informationTypeID*)

Return Value:

string

The code defined by the Feature Catalogue for the information type of the information type instance.

Parameters:

informationTypeID: string

Used by the host to uniquely identify an information type instance.

Remarks:

Instructs the host to return the information type code for the information type instance identified by *informationTypeID*.

13-8.2.1.5 string[] HostFeatureGetSimpleAttribute(string *featureID*, path *path*, string *attributeCode*)

Return Value:

string[]

The textual representation of each attribute value, as described in clause 13-8.1. An array is returned even if the attribute has a single value.

Parameters:

featureID: string

Used by the host to uniquely identify a feature instance.

path: path

An attribute path as described in clause 13-6.2

attributeCode: string

One of the attribute codes defined in the Feature Catalogue for the feature type identified by *featureID*.

Remarks:

Instructs the host to perform a simple attribute lookup on the attribute *attributeCode* at the path *path* for the feature instance identified by *featureID*. An empty array is returned if the requested attribute is not present.

13-8.2.1.6 integer HostFeatureGetComplexAttributeCount(string *featureID*, path *path*, string *attributeCode*)

Return Value:

integer

The number of matching complex attributes that exist at the path for the feature instance.

Parameters:

featureID: string

Used by the host to uniquely identify a feature instance.

path: path

An attribute path as described in clause 13-6.2.

attributeCode: string

One of the attribute codes defined in the Feature Catalogue for the feature type identified by *featureID*.

Remarks:

Instructs the host to return the number of attributes matching *attributeCode* at the given attribute path for the given feature instance. The given path will always be valid for the feature instance. The returned integer can be zero.

13-8.2.1.7 SpatialAssociation[] HostFeatureGetSpatialAssociations(string *featureID*)

Return Value:

SpatialAssociation[]

A Lua array containing all of the spatial associations for the feature instance represented by *featureID.*

Parameters:

featureID: string

Used by the host to uniquely identify a feature instance.

Remarks:

Instructs the host to return an array containing the spatial associations for the given feature instance. For each spatial association the feature contains, the host calls the standard catalogue function *CreateSpatialAssociation* to create the *SpatialAssociation* object.

The host should return an empty array if the feature has no spatial associations.

13-8.2.1.8 string[] HostFeatureGetAssociatedFeatureIDs(string *featureID*, string *associationCode*, variant *roleCode*)

Return Value:

string[]

A Lua array containing the associated features IDs.

Parameters:

featureID: string

Used by the host to uniquely identify a feature instance.

associationCode: string

Code for requested association as defined by the Feature Catalogue.

roleCode: string or nil

Code for requested role as defined by the Feature Catalogue. Can be nil if *associationCode* by itself is enough to specify the association or if all roles defined by *associationCode* are desired.

Remarks:

When called, the host returns an array containing the feature IDs associated with the given feature instance that match *associationCode* and *roleCode*. If no matches are found the host returns an empty array.

The *roleCode* may be nil, in which case only the *associationCode* should be used for lookup.

13-8.2.1.9 string[] HostFeatureGetAssociatedInformationIDs(string *featureID*, string *associationCode*, variant *roleCode*)

Return Value:

string[]

A Lua array containing the associated information IDs.

<u>Parameters:</u>

featureID: string

Used by the host to uniquely identify a feature instance.

associationCode: string

Code for requested association as defined by the Feature Catalogue.

roleCode: string or nil

Code for requested role as defined by the Feature Catalogue. Can be nil if *associationCode* by itself is enough to specify the association or if all roles defined by *associationCode* are desired.

<u>Remarks:</u>

When called, the host returns an array containing the information IDs associated with the given feature instance that match *associationCode* and *roleCode*. If no matches are found the host returns an empty array.

The roleCode may be nil, in which case only the *associationCode* is used for lookup.

13-8.2.1.10 string[] HostInformationGetAssociatedInformationIDs(string *informationID*, string *associationCode*, variant *roleCode*)

<u>Return Value:</u>

string[]

A Lua array containing the associated information IDs.

<u>Parameters:</u>

informationID: string

Used by the host to uniquely identify an information instance.

associationCode: string

Code for requested association as defined by the Feature Catalogue.

roleCode: string or nil

Code for requested role as defined by the Feature Catalogue. Can be nil if *associationCode* by itself is enough to specify the association or if all roles defined by *associationCode* are desired.

<u>Remarks:</u>

When called, the host returns an array containing the information IDs associated with the given information instance that match *associationCode* and *roleCode*. If no matches are found the host returns an empty array.

The roleCode may be nil, in which case only the *associationCode* is used for lookup.

13-8.2.1.11 string[] HostGetSpatialIDs()

<u>Return Value:</u>

string[]

A Lua array containing all of the spatial IDs in the dataset.

<u>Remarks:</u>

Instructs the host to return all spatial IDs relevant to the current scripting catalogue operation. This would typically be all of the spatial objects in an *S100_Dataset* or *S100_DataCoverage*.

As discussed in clause 13-8, the host is responsible for ensuring each spatial ID uniquely identifies a single spatial instance among all product types and datasets to be used during the current scripting session.

13-8.2.1.12 Spatial HostGetSpatial(string *spatialID*)

Return Value:

Spatial

A spatial object created via a standard catalogue function as listed in the remarks*.*

Parameters:

spatialID: string

Used by the host to uniquely identify a spatial.

Remarks:

Queries the host for a given spatial.

The host returns a spatial object created by one of the standard catalogue functions defined in clause 13-8.1.1.

13-8.2.1.13 variant HostSpatialGetAssociatedInformationIDs(string *spatialID*, string *associationCode*, variant *roleCode*)

Return Value:

nil

The information association is not valid for this spatial.

String[]

A Lua array containing the associated information IDs*.*

Parameters:

spatialID: string

Used by the host to uniquely identify a spatial.

associationCode: string

Code for requested association as defined by the feature catalogue.

roleCode: string or nil

Code for requested role as defined by the feature catalogue. Can be nil if *associationCode* by itself is enough to specify the association or if all roles defined by *associationCode* are desired.

Remarks:

When called, the host returns an array containing the information IDs for the given spatial instance that match *associationCode* and *roleCode*. If the information association is not valid for this feature according to the feature catalogue, the host returns nil. If no matches are found the host returns an empty array.

The *roleCode* may be nil, in which case only the *associationCode* is used for lookup.

13-8.2.1.14 string[] HostSpatialGetAssociatedFeatureIDs(string *spatialID*)

Return Value:

string[]

A Lua array containing the requested associated feature IDs for the spatial identified by *spatialID.*

Nil

No features are associated to the spatial identified by *spatialID.*

Parameters:

spatialID: string

Used by the host to uniquely identify a spatial.

Remarks:

When called, the host returns an array of all feature instances that reference the given spatial. A feature instance is considered to be associated to a spatial either directly through the spatial associations on the feature, or indirectly in the case of curves referenced by composite curves or surface rings.

13-8.2.1.15 string[] HostInformationTypeGetSimpleAttribute(string *informationTypeID*, path *path*, string *attributeCode*)

Return Value:

string[] or *nil*

The textual representation of each attribute value, as described in clause 13-8.1. An array is returned even if the attribute has a single value. The host should return nil if the requested attribute is not present.

Parameters:

informationTypeID: string

Used by the host to uniquely identify an information instance.

path: path

An attribute path as defined in clause 13-6.2.

attributeCode: string

One of the attribute codes defined in the Feature Catalogue for the information type identified by *informationTypeID*.

Remarks:

Instructs the host to perform a simple attribute lookup on the attribute *attributeCode* at the indicated *path* for the information instance identified by *informationTypeID*. Nil is returned if the requested attribute is not present.

13-8.2.1.16 integer HostInformationTypeGetComplexAttributeCount(string *informationTypeID*, path *path*, string *attributeCode*)

Return Value:

integer

The number of matching complex attributes that exist at the path for the information instance.

Parameters:

informationTypeID: string

Used by the host to uniquely identify an information instance.

path: path

An attribute path as described in clause 13-6.2.

attributeCode: string

One of the attribute codes defined in the Feature Catalogue for the information type identified by *informationTypeID*.

Remarks:

Instructs the host to return the number of attributes matching *attributeCode* at the given attribute path for the given information instance. The given path will always be valid for the information instance. The returned integer can be zero.

13-8.2.2 Type information access functions

These functions allow the scripting environment to query the type information for any entity from any dataset. The type information provided by the host must match the information from the relevant feature catalogue.

13-8.2.2.1 string[] HostGetFeatureTypeCodes()

Return Value:

string[]

Array containing all feature type codes as defined in the Feature Catalogue.

Remarks:

13-8.2.2.2 string[] HostGetInformationTypeCodes()

Return Value:

string[]

Array containing all information type codes as defined in the Feature Catalogue.

Remarks:

13-8.2.2.3 string[] HostGetSimpleAttributeTypeCodes()

Return Value:

string[]

Array containing all simple attribute type codes as defined in the Feature Catalogue.

Remarks:

13-8.2.2.4 string[] HostGetComplexAttributeTypeCodes()

Return Value:

string[]

Array containing all complex attribute type codes as defined in the Feature Catalogue.

Remarks:

13-8.2.2.5 string[] HostGetRoleTypeCodes()

Return Value:

string[]

Array containing all role type codes as defined in the Feature Catalogue.

Remarks:

13-8.2.2.6 string[] HostGetInformationAssociationTypeCodes()

Return Value:

string[]

Array containing all information association type codes as defined in the Feature Catalogue.

Remarks:

13-8.2.2.7 string[] HostGetFeatureAssociationTypeCodes()

Return Value:

string[]

Array containing all feature association type codes as defined in the Feature Catalogue.

<u>Remarks</u>:

13-8.2.2.8 FeatureType HostGetFeatureTypeInfo(string *featureCode*)

<u>Return Value:</u>

FeatureType

Lua data structure created by the *CreateFeatureType*() function.

<u>Parameters:</u>

featureCode: string

Feature code matching an entry in the Feature Catalogue.

<u>Remarks</u>:

13-8.2.2.9 InformationType HostGetInformationTypeInfo(string *informationCode*)

<u>Return Value:</u>

InformationType

Lua data structure created by the *CreateInformationType*() function.

<u>Parameters:</u>

informationCode: string

Information code matching an entry in the Feature Catalogue.

<u>Remarks</u>:

13-8.2.2.10 SimpleAttribute HostGetSimpleAttributeTypeInfo(string *attributeCode*)

<u>Return Value:</u>

SimpleAttribute

Lua data structure created by the *CreateSimpleAttribute*() function.

<u>Parameters:</u>

attributeCode: string

Simple attribute code matching an entry in the Feature Catalogue.

<u>Remarks</u>:

13-8.2.2.11 ComplexAttribute HostGetComplexAttributeTypeInfo(string *attributeCode*)

<u>Return Value:</u>

ComplexAttribute

Lua data structure created by the *CreateComplexAttribute*() function.

<u>Parameters:</u>

attributeCode: string

Complex attribute code matching an entry in the Feature Catalogue.

<u>Remarks</u>:

13-8.2.2.12 Role HostGetRoleTypeInfo(string roleCode)

<u>Return Value:</u>

Role

Lua data structure created by the *CreateRole* function

<u>Parameters:</u>

roleCode: string

Role code matching an entry in the feature catalogue.

Remarks:

13-8.2.2.13 InformationAssociation HostGetInformationAssociationTypeInfo(string informationAssociationCode)

Return Value:

InformationAssociation

Lua data structure created by the *CreateInformationAssociation* function

Parameters:

informationAssociationCode: string

Information association code matching an entry in the feature catalogue.

Remarks:

13-8.2.2.14 FeatureAssociation HostGetFeatureAssociationTypeInfo(string featureAssociationCode)

Return Value:

FeatureAssociation

Lua data structure created by the *CreateFeatureAssociation* function

Parameters:

featureAssociationCode: string

Feature association code matching an entry in the feature catalogue.

Remarks:

13-8.2.3 Spatial operations functions

These functions allow the scripting environment to perform relational tests and operations on spatial elements.

The host must implement the functions described on the following pages to provide the scripting environment with the ability to relate spatial entities to one another.

13-8.2.3.1 boolean HostSpatialRelate(string *spatialID1*, string *spatialID2*, string *intersectionPatternMatrix*)

Return Value:

boolean

Returns *true* if the geometries represented by the two spatials are related as specified in the DE-9IM matrix.

Parameters:

spatialID1: string

Used by the host to uniquely identify a spatial instance.

spatialID2: string

Used by the host to uniquely identify a spatial instance.

intersectionPatternMatrix: string

DE-9IM intersection matrix expressed as nine characters in row major order. For example, when testing for overlap between two areas: “T*T***T**”

Remarks:

Spatially relates the geometries represented by *spatialID1* and *spatialID2* using the DE-9IM intersection specified via the *intersectionPatternMatrix* string.

For details on DE-9IM string representation refer to ISO 19125-1:2004, *Geographic information -- Simple feature access -- Part 1: Common architecture, section 6.1.14.2 The Dimensionally Extended Nine-Intersection Model (DE-9IM)*.

13-8.2.4 Debugger support functions

These functions allow the scripting environment to interact with a debugger which may be running on the host. A debugger may be desired as an aide in developing the required standard host functions.

Host implementation of the debugger support functions is optional. Scripts will execute normally regardless of whether the host implements these functions.

13-8.2.4.1 void HostDebuggerEntry(string *debugAction*, variant *parameters*)

Return Value:

None

Parameters:

debugAction: string

Indicates the requested debugger action:

break – Pause execution of the script.

trace – Display a string in the debugging console, as provided in the first parameter.

start_performance – Begin line by line profiling of the script code.

stop_performance – Stop line by line profiling of the script code. The name of the performance counter is in the first parameter.

first_chance_error – Notifies the host of an impending error function call in the script. The parameter is the message passed to the error function. The second parameter is the depth passed to the error function.

parameters: variant

Zero or more parameters for use by the debug action.

Remarks:

Host implementation of this function is optional.

S-100 – Part 14

Online Data Exchange

Page intentionally left blank

Contents

Page intentionally left blank

14-1 Scope

This Part describes the components and processes needed to specify an online exchange of information. It could be a set of data or data which may have a continuous nature. The latter is also known as “streaming data”, wherein the data requires a more dynamic information flow to be available; that is, beyond that found with the exchange of static datasets mostly handled as files.

14-2 Normative references

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

IEC 61162, *Maritime navigation and radiocommunication equipment and systems - Digital interfaces – Part 1: Single tanker and multiple instances*

IEC 61174, *Maritime navigation and radiocommunication equipment and systems - Electronic chart display and information system (ECDIS) - Operational and performance requirements, methods of testing and required test results*

ISO/IEC 8211:1994, *Specification for a data descriptive file for information interchange Structure implementations*

ISO/IEC 7498, *Information processing systems – Open Systems interconnection – Basic Reference Model*

ISO/IEC 8859-1:1998, *Information technology – 8-bit single-byte coded graphic character sets – Part 1: Latin alphabet No. 1*

IHO Draft on S-124 for Maritime Safety Information (http://www.iho.int/mtg_docs/com_wg/CPRNW/S100_NWG/2016/S-124NW-CG-01_2016-Draft_Product_Specification-03.12.2015.zip)

OGC Sensor Observation Service (http://www.opengeospatial.org/standards/sos)

W3C Recommendation “SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)” (https://www.w3.org/TR/soap12/)

W3C Recommendation “Web Services Description Language (WSDL)” (https://www.w3.org/TR/wsdl20/)

14-2.1 Open Systems Interconnection (OSI)

This Part makes references to the ISO/OSI standard reference model for open systems interconnection [ISO/IEC 7498], but it does not adhere to that standard with regard to the exact services provided. The ISO/OSI standard is used as a reference for the naming of the individual layers in the protocol stack (see Figure 14.1).

The following conventions apply:

- with respect to functionality, the protocol definitions cover the session, the presentation and the application layers of the OSI model (the A-profile);
- the protocol requires a set of transport services. The services can possibly be supplied by any number of different transport protocol stacks (T-profiles);
- this Part does not describe the A-profile as layered. This Part merges all the upper three layers of the ISO/OSI model into one protocol;
- this Part refers to the companion standards or user layer as a distinct protocol layer on top of the application layer.

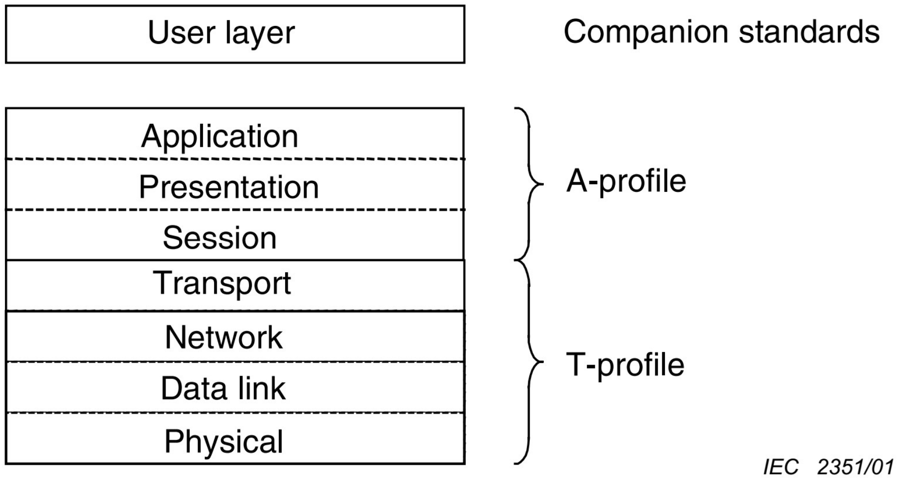


Figure 14-1 — Protocol Layering

14-3 Introduction

Online data exchange between applications/devices will follow different communication patterns to support the variety of maritime operational needs.

Multiple clients can interact with a service to interchange data which is modelled with S-100. It can be distinguished between unidirectional message streams and interactive information exchange.

Context for a communication can be given by using the concept of session oriented communication. Therefore, the communication between distinguished communication partners can be assigned to a logical entity – a session. This allows to store metadata for the interactions assigned to the session.

The means of communication for the use of a service should be defined in a communication stack. Specifying a communication stack will ensure that communication for the service is harmonized and will make implementation easier.

14-3.1 Communication stack

The communication is organized by a stack as defined by the ISO-OSI Reference Model and cover at the A-profile for example:

- Session protocols (for example WSDL, SOAP, REST, SoS) to define message types;
- Encoding and compression (for example GML, XML, ISO8211, HDF, ….) to serialize data;
- Communication protocol (for example HTTP) with encryption (for example HTTPS) to define interaction between gateways;
- Transportation Layer (for example TCP/IP) with encryption (for example SSL) to define transportation node between gateways.

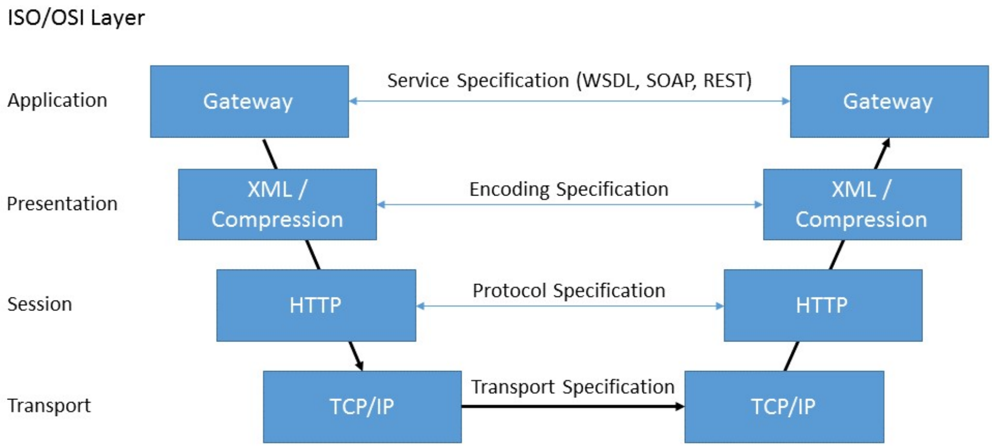


Figure 14-2 — Communication Stack

This Part only addresses the concepts in the application and the presentation layer. The lower layers covering the T-Profile are out of scope of S-100. This could be Internet Protocol or VDES based for example.

14-4 Session oriented communication

To define the context for information exchange the concept of a session shall be used.

A session oriented service typically contains three components, each handling other types of data:

- Session component: Describing the handling of the session data (service request, service response, login, login response, logout).
- Service component: Describing the information to maintain the service (for example keep alive messages, service status).
- Data component: Describing the data itself; for example Vessel Traffic Image data (objects).

Further Metadata required for each component can be detailed in a Product Specification.

In a session oriented service the interfaces are point-to-point connections between client and server. Client and server manage the session (see Figure 14-3) and exchange information bi-directionally. The service description should contain an interaction model. The interaction model should describe the life span of a session (initiation, maintenance and termination of the session).

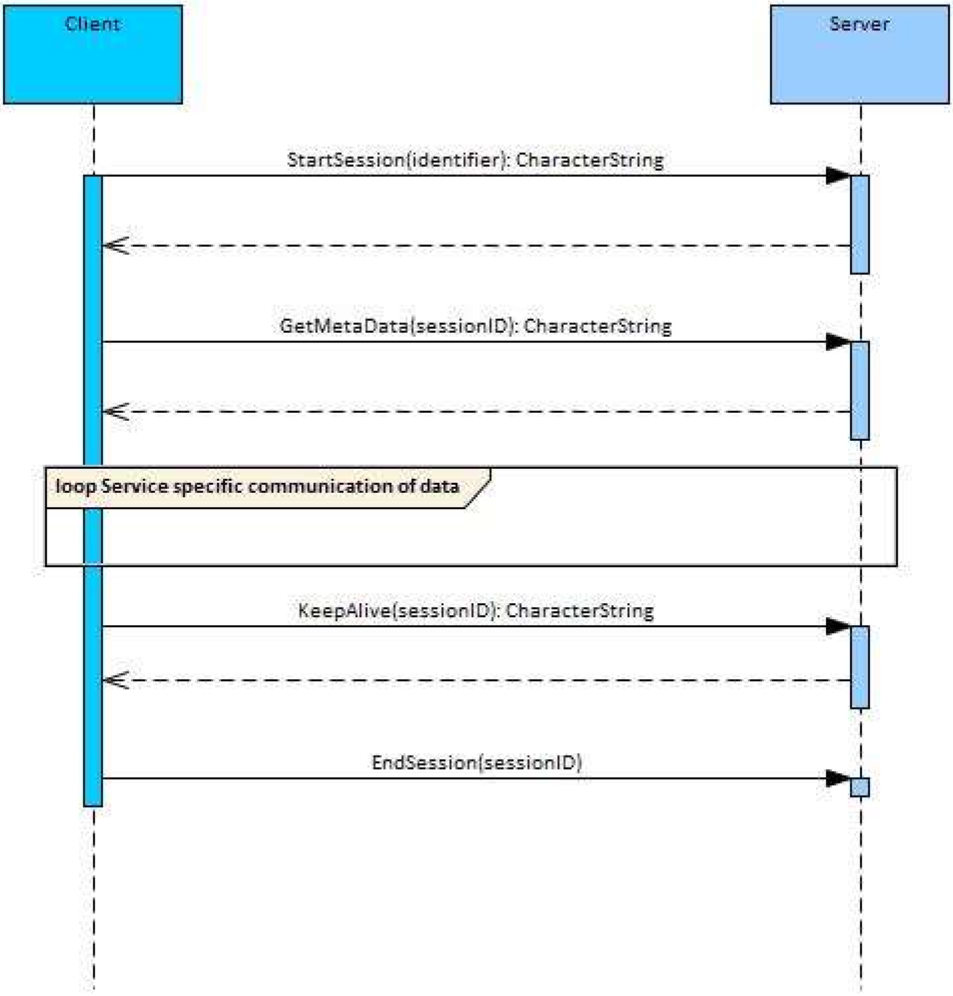


Figure 14-3 — Example of session interaction model

For each element in the interaction model a detailed description shall be provided in the Product Specification of the service. This is to ensure that the service interaction is harmonized and reliable. For example, a description of the protocol used in a service may provide sufficient feedback to ensure full reception of the data, if this is essential for the service.

For each service using the session concept interactions can be defined. For example the following messages:

- Initiate the Session
    - Initiate and confirm Sessions
- Maintenance of Session
    - Keep alive messages
- Termination of the Session
    - Closing Session Request

14-5 Session-less interactive communication

Interactive communication is broadly used in application to application data exchange. Mostly the client server communication pattern is applied. Clients initiate communication with a server and both partners exchange messages as (defined) sets of data.

Following the concepts of stateless communication paradigms a session-less message exchange requires an encapsulation of all relevant information within a request. Based solely on this information, the server shall be able to formulate an appropriate response. Metadata will either be part of this response or shall be provided within the service specification. All operations are service-specific and are therefore not considered here.

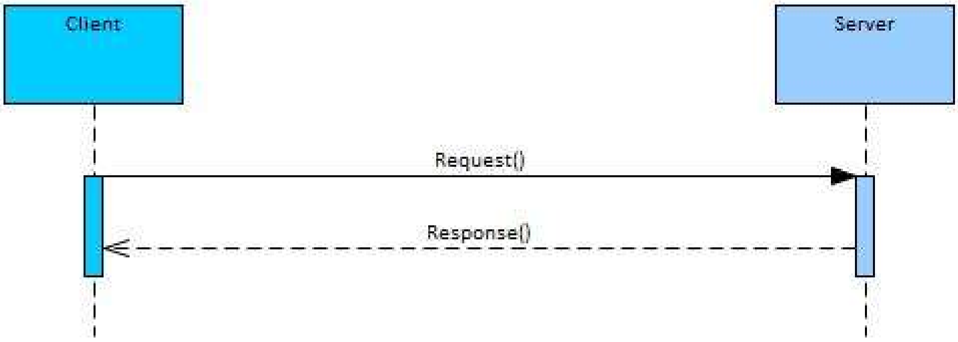


Figure 14-4 — Session-less client-server communication

14-6 Message streams

Message streams are a unidirectional flow of messages containing well-defined sets of data. The used communication medium can ensure sequence and completeness of the message stream.

Contrary to the session concept broadcasted messages are mostly context agnostic. It is possible but not necessary that the message stream from the server is triggered by a message from a client. Therefore, clients can broadcast an undirected request for information followed by an undirected answer by a server. An identifier has to be provided to associate a response message to a request. Message stream messages have to include metadata about the transferred datasets.

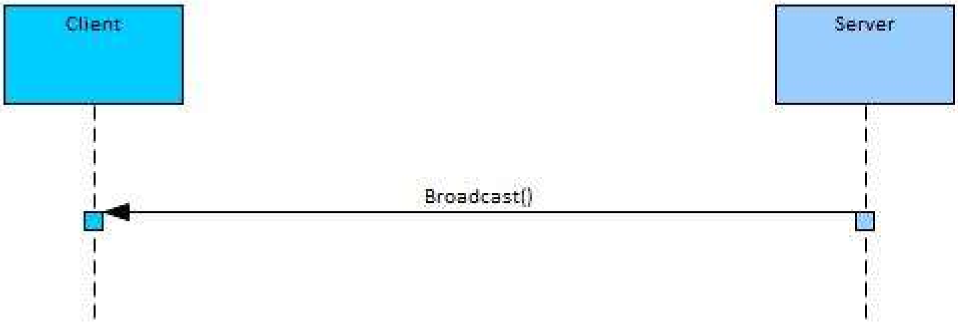


Figure 14-5 — Message streaming communication

14-7 IP based technologies

Generally online data exchange is applicable on different ISO/OSI Service Stacks. For IP based communication it is recommended that S-100 compliant data be communicated using Web Service technologies.

In the following sub-sections two common Web Service technologies are introduced.

14-7.1 SOAP

SOAP relies on the Web Service Definition Language (WSDL) and on XML to provide web services over the internet. The W3C standardized SOAP. SOAP specification can be broadly defined to be consisting of the following three conceptual components: Protocol concepts, Encapsulation concepts and Network concepts. It is designed to support expansion and provides concepts such as:

- WS-Addressing is a specification of transport-neutral mechanisms that allows web services to communicate addressing information. It essentially consists of two parts: a structure for communicating a reference to a Web Service endpoint; and a set of messages addressing properties which associate addressing information with a particular message;
- WS-Policy represents a set of specifications that describe the capabilities and constraints of the security (and other business) policies on intermediaries and end points (for example, required security tokens, supported encryption algorithms, and privacy rules) and how to associate policies with services and end points;
- WS-Security is an extension to SOAP to apply security to Web services;
- WS-Federation is part of the larger Web Services Security framework. WS-Federation defines mechanisms for allowing different security realms to broker information on identities, identity attributes and authentication;
- WS-ReliableMessaging describes a protocol that allows SOAP messages to be reliably delivered between distributed applications in the presence of software component, system, or network failures;
- WS-Coordination describes an extensible framework for providing protocols that coordinate the actions of distributed applications;
- WS-AtomicTransaction consists of protocols and services that together ensure automatic activation, registration, propagation and atomic termination of Web services. The protocols are implemented via the WS-Coordination context management framework and emulate ACID transaction properties

The SOAP message is an XML document consisting of a SOAP-Envelope containing an optional SOAP-Header, the SOAP-Body and optional SOAP-Fault information on errors that occurred while processing a message. The envelope creates the namespace for the message; the optional header can contain meta-data concerning, for example, routing and encryption; and the body contains the data of the message to the SOAP-receiver.

```xml
<?xml version="1.0"?>
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope">
     <s:Header>
     </s:Header>
     <s:Body>
     </s:Body>
     <s:Fault>
     </s:Fault>
</s:Envelope>
```

Using SOAP in the context of S-100 will require using a reference of the Service Definition Model in the SOAP-Header and placing the S100_DataSet into the SOAP-Body. See Appendix B for an example.

14-7.2 REST

REST is acronym for REpresentational State Transfer. It is an architectural style for distributed hypermedia systems and was first presented by Roy Fielding in 2000. REST has six guiding constraints which must be satisfied if an interface needs to be referred as RESTful. These principles are listed below.

Guiding Principles of REST:

- Client–server: By separating the user interface from data storage, REST improves the portability of the user interface across multiple platforms and improves scalability by simplifying the server components.
- Stateless: Each request from client to server must contain all of the information necessary to understand the request, and must not take advantage of any stored context on the server. Session state is therefore kept entirely on the client.

- Cacheable: Cache constraints require that the data within a response to a request be implicitly or explicitly labeled as cacheable or non-cacheable. If a response is cacheable, then a client cache is given the right to reuse that response data for later, equivalent requests.
- Uniform interface: By applying the software engineering principle of generality to the component interface, the overall system architecture is simplified and the visibility of interactions is improved. In order to obtain a uniform interface, multiple architectural constraints are needed to guide the behavior of components. REST is defined by four interface constraints: identification of resources; manipulation of resources through representations; self-descriptive messages; and hypermedia as the engine of application state.
- Layered system: The layered system style allows an architecture to be composed of hierarchical layers by constraining component behavior such that each component cannot “see” beyond the immediate layer with which they are interacting.
- Code on demand (optional): REST allows client functionality to be extended by downloading and executing code in the form of applets or scripts. This simplifies clients by reducing the number of features required to be pre-implemented.

The key abstraction of information in REST is a resource. Any information that can be named can be a resource: a document or image; a temporal service; a collection of other resources; a non-virtual object (for example a person); and so on. REST uses a resource identifier to identify the particular resource involved in an interaction between components.

14-8 Service definition model

In Figure 14-6 the service definition model is shown. It defines how to describe the service operations in a generic way. The central part of the model is the S100_OC_ServiceMetaData class. This class defines all information required to implement and use a service. Therefore it references an S100_FC_FeatureCatalogue, which contains all necessary metadata about the datasets exchanged via the service API. This API is defined by one or more interface definitions (by using the S100_OC_ServiceInterface Class). They are composed of a set of operations which are represented in two ways:

1. A formal description: Each of the Operations shall be described in a technology agnostic way, specifying the parameters for the operation as well as its results. A S100_OC_ParameterBinding is a buildup of a direction that defines whether the parameter is read only, write only or both, by the service.

   An additional S100_OC_ParameterBinding (direction: return) specifies the result data type of an operation.

2. A technology dependent description: Each S100_OC_ServiceInterface is composed of a technology identifier (REST, SOAP, etc.) and one or more external technology dependent description files, referenced via the interfaceDescription URLs. In addition, the S100_OC_ServiceInterface can specify the encoding of the data, in case this is not defined through the used technology. When utilized, the encoding attribute has to define the name of the used encoding, for example ISO8211, GML as specified for S100, etc. While these encoding attributes applies to the data within the dataset, it can be overwritten by an encoding attribute of the parameter binding. This allows further specifying the content of a parameter value.

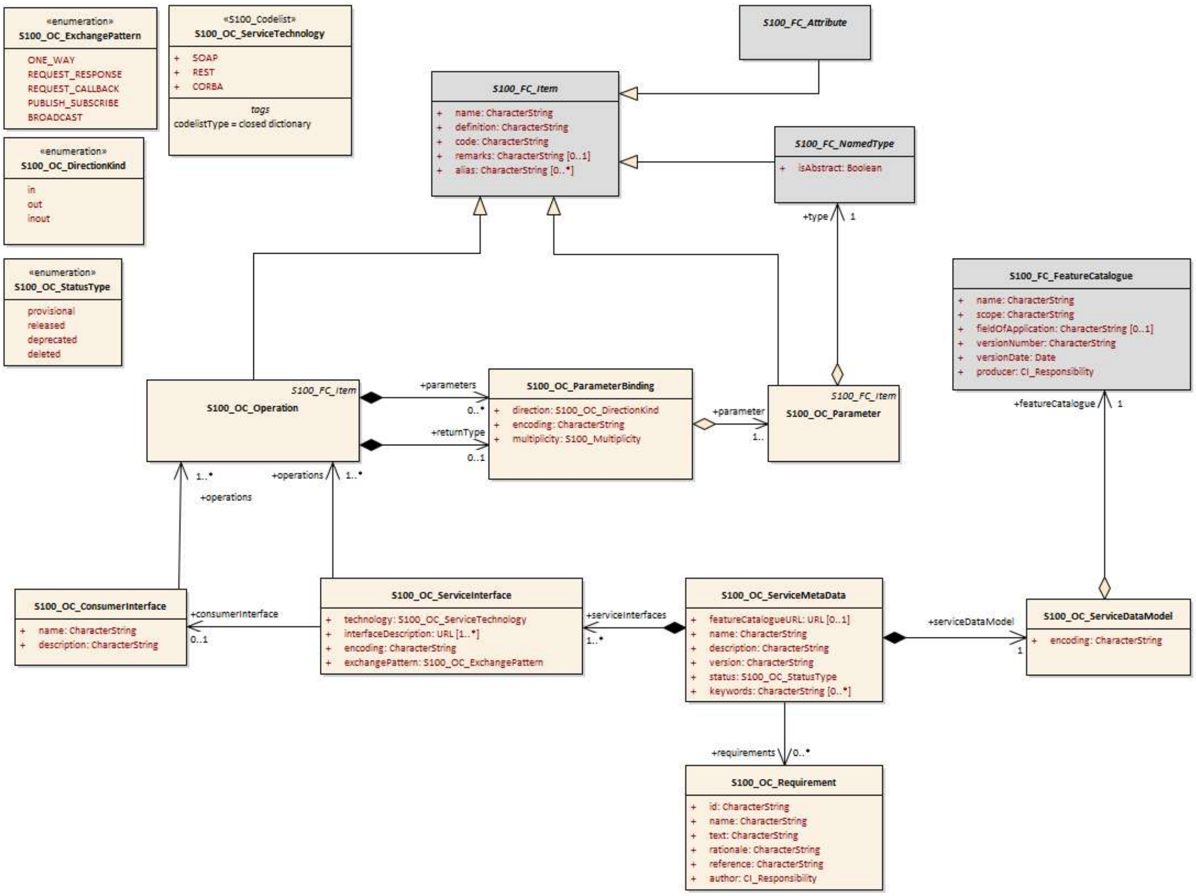


Figure 14-6 — Data model to describe a service

14-8.1 Types

14-8.1.1 S100_OC_ServiceMetaData

Defines all information required to implement the service.

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_OC_ServiceMetaData	Root Entry point to formal describe a service including its interaction models and data products	-	-	-
Composition	serviceDataModel	Describes the logical data model of the service	1	S100_OC_ServiceDatamodel	Mandatory
Composition	serviceInterfaces	Describe the technology agnostic and technology specific interfaces for a service	1..*	S100_OC_ServiceInterface	Mandatory
Attribute	featureCatalogueURL	URL to the used Feature Catalogue. This URL should if possible, point to a machine readable representation of the FeatureCatalogue, referred in Exchange Set	0..1	URL	Mandatory
Association	requirements	Refers to requirements specifications for the service. Business requirements, functional and non-functional requirements should be listed here. At least one requirement shall be given	0..*	S100_OC_Requirement	
Attribute	name	The human readable service name. The service name shall be at maximum a one-line brief label for the service. Newer versions of the same service specification shall not change the name	0..1	CharacterString	
Attribute	description	A human readable short description of the service. The description shall contain an abstract of what a service implementing this specification would do	0..1	CharacterString	
Attribute	version	Version of the service specification. A service specification is uniquely identified by its name and version. Any change in the service data model or in the service interface definition requires a new version of the service specification	0..1	CharacterString	
Attribute	status	Status of the service specification	0..1	S100_OC_StatusType	
Attribute	keywords	A list of keywords associated with the service	0..*	CharacterString	

14-8.1.2 S100_OC_ServiceInterface

Specifies the given technology, as well as a reference to a technology dependent description for that interface. The interfaceDescription has to point to a technology dependent interface definition file that matches the operations, defined through the “operations” aggregation. In addition, the ServiceInterface can specify the encoding of the data, in case this is not defined by the used technology.

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_OC_ServiceInterface	Describe the technology agnostic and technology specific interfaces for an service	-	-	-
Attribute	technology	Used technology	1	S100_OC_Service Technology	Mandatory
Attribute	interfaceDescription	Technology depended definition file for the operations. Has to match with the “operations” aggregation	1..*	URL	Mandatory
Attribute	encoding	Encoding of the data sets used in this interfaceDefinition. Has to be set if the encoding is not defined through the used technology	0..1	CharacterString	Conditional, has to be set if the encoding is not defined through the used technology
Attribute	exchangePattern	Describes the type of interaction that is supported	1	S100_OC_ExchangePattern	Mandatory
Association	operations	Technology agnostic description of operations provided by this service	1..*	S100_OC_Operation	Mandatory
Association	consumerInterface	Optional reference to an interface definition that shall be provided by the service consumer to complement the service interface. Especially if a publish/subscribe service interface is designed, it is necessary to describe what the service expects to be available on the subscriber side	0..1	S100_OC_ConsumerInterface	Optional

14-8.1.3 S100_OC_Operation

Defines the operations possible on the specified service in a technology agnostic way. Specifies the Parameters as well as the results of the operations (see S100_OC_ParameterBinding).

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_OC_Operation	Specifies on operation that can be performed by a service	-	-	-
Generalisation	-	Use the same description methodology for Features, Attributes, … and Operations	1	S100_FC_Item	Mandatory
Composition	parameters	List of owned parameter bindings. Its obligation is defined by the semantic of the operation, for example if input / output is required	0..*	S100_OC_ParameterBinding	
Composition	returnType	Parameter to deliver results of an operation back to the caller	0..1	S100_OC_ParameterBinding	

14-8.1.4 S100_OC_Parameter

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_OC_Parameter		-	-	-

Further information regarding S100_OC_Parameter will be included in a future Edition of S-100.

14-8.1.5 S100_OC_ParameterBinding

Assigns an S100_OC_Parameter to an Operation. It follows the S-100 concept for the assignment and restriction of attributes and supplements it with the definition of a direction (see section 14-8.2).

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_OC_ParameterBinding	Class that is used to describe how an Attribute can be bound to an operation	-	-	-
Attribute	direction	Specifies how the operation uses the parameter	1	S100_OC_DirectionKind	Mandatory
Attribute	encoding	If set, this attribute specifies the encoding used for this parameter. If not set, the technology dependent encoding is used	0..1	CharacterString	
Attribute	multiplicity	Minimum and maximum number of provided instances, where the maximum number may be infinitive. If no multiplicity is provided a multiplicity of 1 is assumed	0..1	S100_Multiplicity	
Aggregation	parameter	Used to describe the type of the parameter	1..*	S100_OC_Parameter	

14-8.1.6 S100_OC_Requirement

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_OC_Requirement	A requirement that the service shall fulfil	-	-	-
Attribute	id	Globally unique requirement identification	1	CharacterString	Mandatory
Attribute	name	Human readable requirement name/summary. Shall not be longer than one line	1	CharacterString	Mandatory
Attribute	text	The human readable requirement text. Usually formulated in form of a 'shall'-statement	1	CharacterString	Mandatory

Attribute	rationale	Rationale for this requirement. Textual explanation of why this requirement exists. Provides background information about the need of the service	1	CharacterString	Mandatory
Attribute	Reference	Optional information about where the requirement was originally stated. If the requirement comes from external documents, this attribute shall refer to this source	0..1	CharacterString	Optional
Attribute	Author	Optional reference(s) to administrative information about the author(s) of the requirement	0..1	CI_Responsibility	Optional

14-8.1.7 S100_OC_ConsumerInferface

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_OC_ConsumerInterface	Interface specification that is expected to be provided by the service consumer. For example, if a request/callback service interface is designed, it is necessary to describe the interface the service expects on the client side	-	-	-
Attribute	Name	Human readable interface name. The name shall be no longer than one line	1	CharacterString	Mandatory
Attribute	description	Human readable description of the interface	1	CharacterString	Mandatory
Association	operations	Refers to the specification of service operations supported by the consumer interface	1..*	S100_OC_Operation	Mandatory

14-8.1.8 S100_OC_ServiceDataModel

Role Name	Name	Description	Mult	Type	Remarks
Aggregation	featureCatalogue	Describes the service data model	1	S100_FC_Feature Cataogue	-
Attribute	encoding	The encoding of the machine-readable representation of the featureCatalogue	0..1	CharacterString	This refers to the Feature Catalogue provided via the featureCatalogueURL in S100_OC_ServiceMetaData

14-8.2 Codelists and enumerations

14-8.2.1 S100_OC_ServiceTechnology

Item	Name	Description	Mult	Type	Remarks
S100_CodeList	S100_OC_ServiceTechnology	List of commonly used service (description / implementation) Technologies	-	-	-
Literal	SOAP	Simple Object Access Protocol	-	-	SOAP Version 1.2 2007, W3C (https://www.w3.org/TR/soap12-part1)
Literal	REST	Representational State Transfer	-	-	Fielding, Roy Thomas (2000). Fielding Dissertation: CHAPTER 5: Representational State Transfer (REST) (uci.edu)
Literal	CORBA	Common Object Request Broker Architecture	-	-	Object Management Group (OMG) Feb 2021 v3.4 (https://www.omg.org/spec/CORBA/)

14-8.2.2 S100_OC_DirectionKind

Item	Name	Description	Mult	Type	Remarks
Enumeration	S100_OC_DirectionKind	Describes how an operation uses an parameter	-	-	-
Literal	in	In(put) parameters can only be read by the owning operation but they will never be changed	-	-	-
Literal	out	Out(put) parameters can be used by the owning operation to store additional information for the caller, their initial content will neither be read nor removed (cleared)	-	-	-
Literal	inout	In(put)/Out(put) parameters can be used by the owning operation to store additional information for the caller, however the content of those parameters also affects the operations execution	-	-	-

14-8.2.3 S100_OC_StatusType

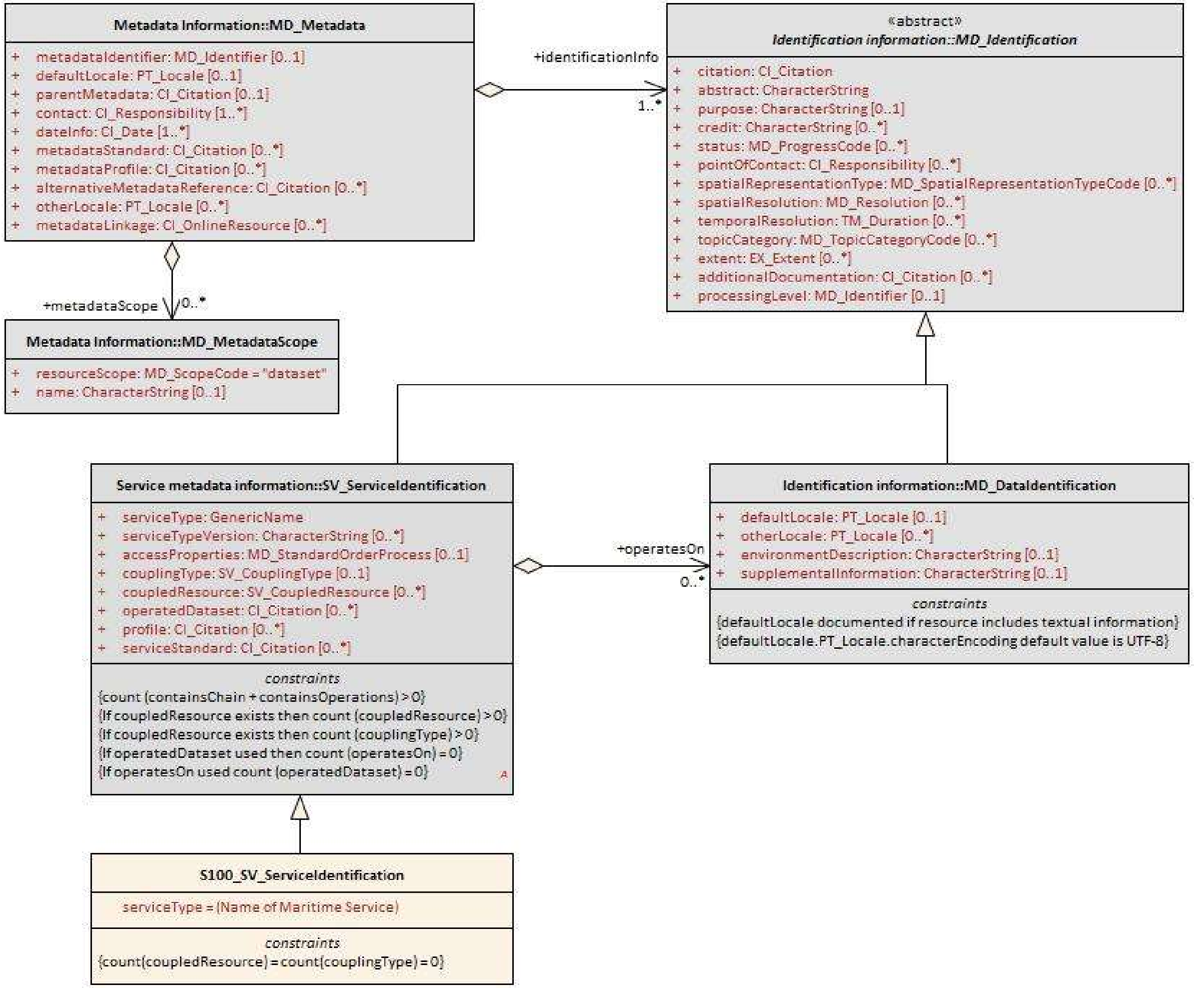
Item	Name	Description	Mult	Type	Remarks
Enumeration	S100_OC_StatusType	Describes the status of a service specification, design or instance	-	-	-
Literal	provisional	The service specification/design is not officially released, the service instance is available, but not in official operation	-	-	-
Literal	released	The service specification/design/instance is officially released	-	-	-

Literal	deprecated	The service specification/design/instance is still available, but end of life is already envisaged	-	-	-
Literal	deleted	The service specification/design/instance is not available any more	-	-	-

14-8.2.4 S100_OC_ExchangePattern

Item	Name	Description	Mult	Type	Remarks
Enumeration	S100_OC_ExchangePattern	Defines operation processing types	-	-	-
Literal	ONE_WAY	Data are sent in one direction, from service consumer to service provider, without confirmation	-	-	-
Literal	REQUEST_RESPONSE	Service consumer sends request to service provider and expects to receive a response from the service provider	-	-	-
Literal	REQUEST_CALLBACK	(asynchronous REQUEST_RESPONSE) Service consumer sends a request to service provider; response is provided asynchronously in an independent call to the service	-	-	-
Literal	PUBLISH_SUBSCRIBE	Service consumer subscribes at service provider for receiving publications sent out by the service provider	-	-	-
Literal	BROADCAST	Service provider distributes information independently of any consumers	-	-	-

14-8.3 Service identification



Source: Adapted from ISO 19115-1:2014

Figure 14-7— Service metadata information classes

14-8.3.1 S100_SV_ServiceIdentification

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_SV_ServiceIdentitification	Identification of capabilities which a service provider makes available to a service user through a set of interfaces which define a behaviour	-	-	Specialization of SV_ServiceIdentification (ISO 19115-1) and thereby a specialization of MD_Identification (The ISO attributes coupledResource and couplingType are not used.)
(Inherited properties)	(Inherited from SV_ServiceIdentification.)				
Attribute	serviceType	A service type name	1	Class GenericName	GenericName is an abstract class for all names in a NameSpace. Each instance of a GenericName is either a LocalName or a ScopedName. A LocalName references a local object directly accessible from the NameSpace. A ScopedName is a composite of a LocalName for locating another NameSpace and a GenericName valid in the NameSpace. (ISO 19103). In short: A name that is defined in a namespace. For S-100 services, the recommended namespace is the IALA/IMO/IHO list of Maritime Services (TBD as of May 2018)
Attribute	serviceTypeVersion	The version of the service, supports searching based on the version of serviceType	0..*	CharacterString	
Attribute	accessProperties	Information about the availability of the service, including fees, planned available date and time, ordering instructions, turnaround	0..1	MD_StandardOrderProcess	ISO 19115-1 B.11.5
Attribute	operatedDataset	Provides a reference to the resource on which the service operates	0..*	CI_Citation	For any single resource referenced, only one of operatedDataset or operatesOn is allowed to be documented (not both for the same resource)
Attribute	profile	Profile to which the service adheres	0..*	CI_Citation	Profile of the standard cited in serviceStandard The specification for the data product can be identified here
Attribute	serviceStandard	Standard to which the service adheres	0..*	CI_Citation	For example, citation for OGC WFS, WMS, etc.
Role	operatesOn		0..*	MD_DataIdentification	For any single resource referenced, only one of operatedDataset or operatesOn is allowed to be documented (not both for the same resource)

(Inherited properties)	(Inherited from MD_Identification.) (not shown)

14-9 Communication management data types

The client requests the creation of a session from the service provider that returns a session ID. The subsequent communication, whose operations are not part of these recommendations, is always carried out using the SessionID. A second operation closes the active session. Figure 14-8 below shows this minimum set of Operations. The Operation *GetMetaData* allows to request metadata for the data sets at runtime. KeepAlive is called in order to prevent the session from timing out.

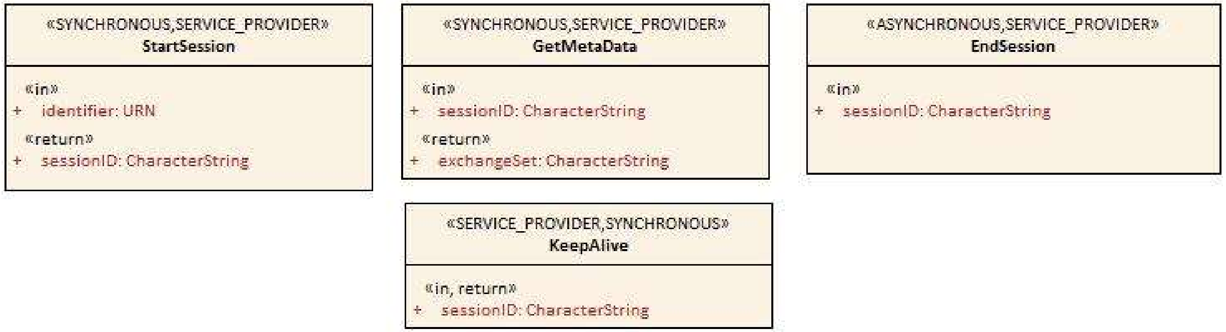


Figure 14-8 — Minimum set of Operations for session based, interactive services

14-9.1 Types

14-9.1.1 StartSession

OPERATIONTYPE: SYNCHRONOUS

OPERATIONOWNER: SERVICE_PROVIDER

Role Name	Name	Description	Mult	Type	Direction
Operation	StartSession	Request to start a new session	-	-	-
Parameter	identifier	World wide unique identification of the requester	1	URN	in

Parameter	sessionID	Service unique identification for the session, that shall match ITU-T Rec X.667 | ISO/IEC 9834-8 If this parameter is empty the login has failed and the parameter "message" contains the reason for failure	1	CharacterString	return

14-9.1.2 EndSession

OPERATIONTYPE: SYNCHRONOUS

OPERATIONOWNER: SERVICE_PROVIDER

Role Name	Name	Description	Mult	Type	Direction
Operation	EndSession	Request to close the session	-	-	-
Parameter	sessionID	Session to be closed, shall match ITU-T Rec X.667 | ISO/IEC 9834-8	1	CharacterString	in

14-9.1.3 GetMetaData

OPERATIONTYPE: SYNCHRONOUS

OPERATIONOWNER: SERVICE_PROVIDER

Role Name	Name	Description	Mult	Type	Direction
Operation	GetMetaData	Request for MetaData of the exchanged datasets	-	-	-
Parameter	sessionID	To identify the active session	1	CharacterString	in
Parameter	exchangeSet	The exchange set describing the datasets.	1	CharacterString	return

14-9.1.4 KeepAlive

OPERATIONTYPE: SYNCHRONOUS

OPERATIONOWNER: SERVICE_PROVIDER

Role Name	Name	Description	Mult	Type	Direction
Operation	KeepAlive	Prevent the session from timing out	-	-	-
Parameter	sessionID	To identify the active session	1	CharacterString	In
Parameter	sessionID	To identify the active session	1	CharacterString	return

Appendix 14-A
Example: Efficient Data Broadcasting
(informative)

This example describes a service providing data broadcasting. The service embeds the data structure given by an external Product Specification. The data items, structured according to the Product Specification are broadcast via a communication medium (for example VDES). Therefore they are serialized and sent in conformity with the IEC/ISO 8211 encoding defined within the standard S-100 (Part 10a).

Figure 14-A-1 shows how to exchange information efficiently. Static data, such as the data structure according to the product definition, is considered part of the service specification (StaticData_ISO8211). Since the client must already know this information in order to use the service, only an exchange of the dynamic data is necessary (DynamicData_ISO8211). The service provider reduces the data set serialized in ISO 8211 by removing all static data that has already been covered within the service specification. The client receives the data and merges it with the static data record. In this way, the entire data set can be reconstructed. The basis for such a concept is the Insert, Delete, and Modify mechanism as described in S-100 Part 10a. Therefore, it is possible to represent both static and dynamic data separately as ISO 8211 compliant.

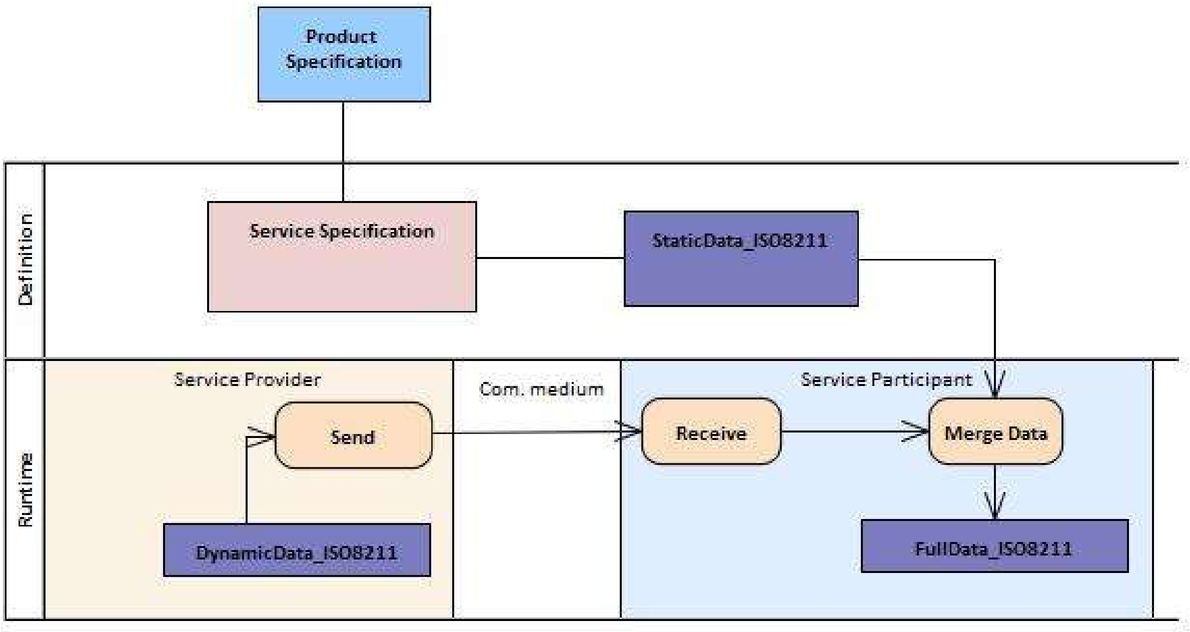


Figure 14-A-1 — Example defining a static data exchange service in a Product Specification

Page intentionally left blank

Appendix 14-B
Example: Session Based Web Service
(informative)

This example describes a session based concept (see clause 14-4) for the transmission of Navigational Warnings. The data structure for such messages is defined in the Product Specification S-124 and will be provided as an XML schema.

The service described here enables a consumer to request messages related to a specific area. At the technological level, SOAP is used. Figure 14-B-1 shows the attribute values of the ServiceInterface. As described in section 14.8, a ServiceInterface contsists of a formal and a technology-specific part. The formal specification of all the necessary operations is shown in Figure 14-B-2.

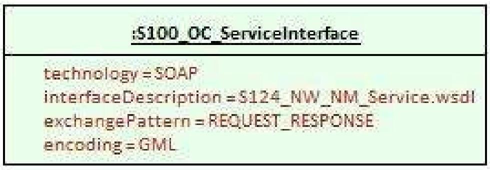


Figure 14-B-1 — ServiceInterface instance values

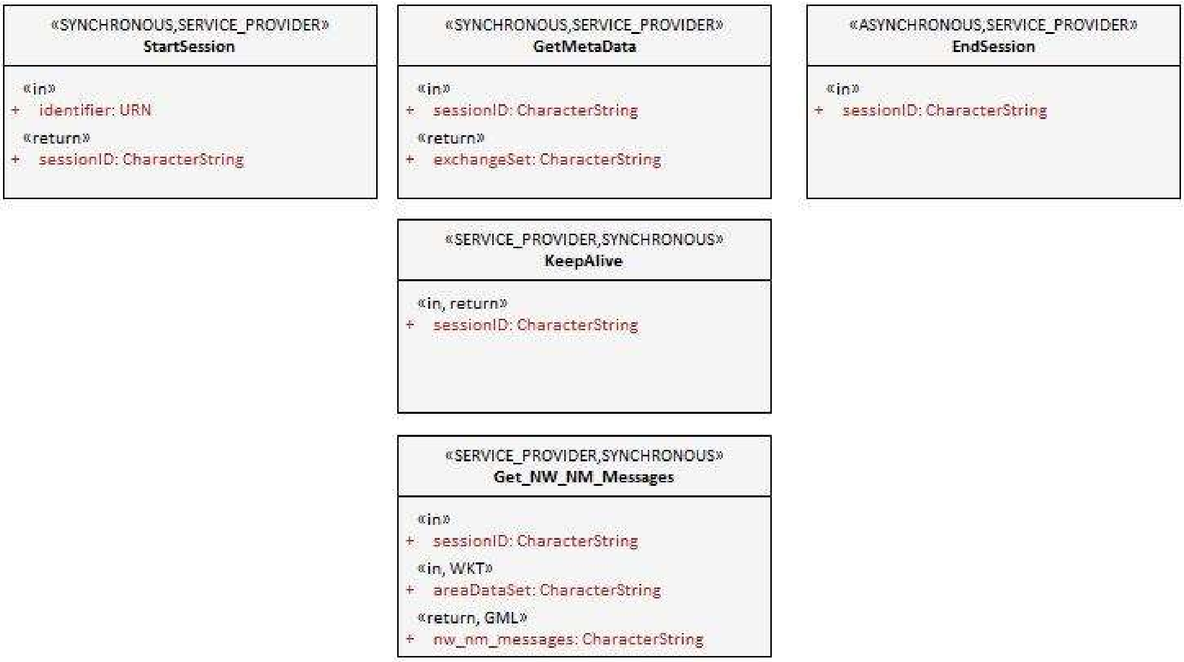


Figure 14-B-2 — NW-NM Service formal definition of the Operations

As defined in the ServiceInterface, the technology-specific part is described by a WSDL file. This is shown below.

Once a client wishes to access Nautical Warnings and Notices to Mariners, it starts a session by using the StartSession operation, to which the Server will reply by issuing a sessionID. The client then starts requesting the messages for a specific area using the Get_NW_NM_Messages operation. The server's response will be the nw_nm_messages data-set, which the client will be able to interpret through the S-124 Product Specification.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<wsdl:definitions xmlns:tns="http://www.example.org/S124_NW_NM_Service/"
        xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema" name="S124_NW_NM_Service"
        targetNamespace="http://www.example.org/S124_NW_NM_Service/">

  <wsdl:types>
        <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
                <xsd:import id="S124.xsd"  schemaLocation="http://www.iho.int/S124/gml/1.0"
              namespace="S124"/>
        </xsd:schema>
  </wsdl:types>
  <wsdl:message name="StartSessionRequest">
         <wsdl:part name="identifier" type="xsd:string" />
  </wsdl:message>
        ...
  <wsdl:message name="Get_NW_NM_Request">
         <wsdl:part name="sessionID" type="xsd:string" />
         <wsdl:part name="areaDataSet" type=" xsd:string " />
  </wsdl:message>
  <wsdl:message name="Get_NW_NM_Response">
         <wsdl:part name="nw_nm_messages" type=" xsd:string " />
  </wsdl:message>
  <wsdl:portType name="S124_NW_NM_Service">
         <wsdl:operation name="StartSession">
                <wsdl:input message="tns:StartSessionRequest" name="" />
                <wsdl:output message="tns:StartSessionResponse" />
         </wsdl:operation>
                ...
         <wsdl:operation name="Get_NW_NM_Messages">
                <wsdl:input message="tns:Get_NW_NM_Request" />
                <wsdl:output message="tns:Get_NW_NM_Response" />
         </wsdl:operation>
  </wsdl:portType>

  <wsdl:binding name="S124_NW_NM_ServiceSOAP" type="tns:S124_NW_NM_Service">
         <soap:binding style="document"
                       transport="http://schemas.xmlsoap.org/soap/http" />
                <wsdl:operation name="StartSession">
                       <soap:operation
                              soapAction="http://www.example.org/S124_NW_NM_Service/StartSession" />
                       <wsdl:input name="">
                              <soap:body use="literal" />
                       </wsdl:input>
                       <wsdl:output>
                              <soap:body use="literal" />
                       </wsdl:output>
                </wsdl:operation>
        </wsdl:binding>
        <wsdl:service name="S124_NW_NM_Service">
                <wsdl:port binding="tns:S124_NW_NM_ServiceSOAP" name="S124_NW_NM_ServiceSOAP">
                       <soap:address location="http://www.example.org/" />
                </wsdl:port>
        </wsdl:service>
</wsdl:definitions>
```

S124_NW_NM_Service.wsdl

Appendix 14-C
Operations
(informative)

Descriptions of the StartSession, EndSession, KeepAlive and GetMetaData Operations can be found in section 14.9 and are therefore not explained here.

14-C.1 Get_NW_NM_Service

OPERATIONTYPE: SYNCHRONOUS

OPERATIONOWNER: SERVICE_PROVIDER

Role Name	Name	Description	Mult	Type	Direction	Encoding
Operation	Get_NW_NM_Messages	Provides NW and NM messages for a specific area	-	-	-	
Parameter	sessionID	To identify the active session	1	Character String	in	
Parameter	areaDataSet	The area definition	0..1	Character String	in	WKT
Parameter	nw_nm_messages	The messages returned for the area	1	Character String	return	GML

This operation uses the additional encoding field for a parameter binding to further specify the content and format of two parameters. That is, the return message will return a CharacterString that uses GML to encode the content of the String and thus defines its meaning. The input parameter “areaDataSet” expects the String to be encoded as Well Known Text geometry, at least if not empty.

Page intentionally left blank

S-100 – Part 15

Data Protection Scheme

Page intentionally left blank

Contents

15-1 Scope

S-100 part 15, later referred to as 'the Data Protection Scheme' or 'Protection Scheme', describes the recommended standard for the protection of hydrographic or spatial information based on the IHO S-100 Universal Hydrographic Data Model. It defines security constructs and operating procedures that must be followed to ensure that the Protection Scheme is operated correctly and to provide specifications that allow participants to build compliant systems and distribute data in a secure and commercially viable manner.

15-2 Normative References

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

FIPS Publication 81, *DES Modes of Operation*, National Institute of Standards and Technology <www.itl.nist.gov/fipspubs/fip81.htm>

FIPS Publication 180-4, *Secure Hash Standard (SHS)* <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf>

FIPS Publication 186, *Digital Signature Standard (DSS)* <www.itl.nist.gov/div897/pubs/fip186.htm>

ISO/IEC 18033-3, *Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers (AES)*

Open SSL Cryptography and SSL/TLS Toolkit <https://www.openssl.org/>

PKCS#10 v1.7, *Certification Request Syntax Specification* <https://tools.ietf.org/html/rfc2986>

RFC 1423, *Privacy Enhancements for Internet Electronic Mail: Part III: Algorithms, Modes and Identifiers* <ftp://ftp.isi.edu/in-notes/rfc1423.txt>

RFC 2451, *The ESP CBC-Mode Cipher Algorithms* <https://tools.ietf.org/html/rfc2451>

RFC 2459 version 3, *Internet X.509 Public-key infrastructure and attribute certificate frameworks* <https://tools.ietf.org/html/rfc2459>

RFC 5651, *Cryptographic Message Syntax (CMS)*, ITU International Telecommunication Union <https://tools.ietf.org/html/rfc5652#section-6.3>

RFC 4647, Base 64 Encoding. <https://datatracker.ietf.org/doc/html/rfc4648#section-4>

OSI networking and system aspects – Abstract Syntax Notation One (ASN.1), ITU International Telecommunication Union <https://www.itu.int/ITU-T/studygroups/com17/languages/X.680-0207.pdf>

X.509 Version 3, *Information Technology – Open Systems Interconnection – The Directory: Authentication Framework*, International Telecommunication Union

15-3 General Description

This Part specifies a method of securing digital nautical, hydrographic and spatial related products and information. The purpose of data protection is threefold:

1. Piracy Protection: To prevent unauthorized use of data by encrypting the product information.
2. Selective Access: To restrict access to only the products that a customer has acquired a license for.
3. Authentication: To provide assurance that the products have come from approved sources.

Piracy protection and selective access are achieved by encrypting the products and providing data permits to decrypt them. Data permits have an expiration date to enable access to the products for a licensed period. Data Servers will encrypt digital products before supplying them to the Data Client. The encrypted products are then decrypted by the end-user system (for example ECDIS/ECS) prior to use. Authentication is provided by means of digital signatures applied to the product files.

The security scheme does not address how the product information is protected once it is within an end-user application. This is the responsibility of the Original Equipment Manufacturers (OEMs).

The Scheme enables the mass distribution of protected datasets on hard media which can then be accessed and used by all customers with a valid license containing a set of data permits. Selective access to individual products is supported by providing users with a licensed set of data permits containing the encrypted dataset keys. This license is created using a unique hardware identifier of the target system and is unique to each Data Client. Consequently licenses cannot be exchanged between individual Data Clients.

The Protection Scheme is designed for file based transfer of data between parties. Stream based transfer may use different methodologies. Data streaming is presented in S-100 Part 14. The S-100 Protection Scheme described in this Part is bound to the value “S100p15” in the protectionScheme element of the CATALOG.XML Exchange Set Catalogue.

The Scheme uses an optional compression algorithm to reduce the size of the dataset. Unencrypted product files contain many repeating patterns of information; for example coordinate information. Compression is therefore always applied before the product file is encrypted and uncompressed after the corresponding decryption on the data client system.

15-4 Participants in the Protection Scheme

There are several types of users of the Scheme; these are as follows:

- The Scheme Administrator (SA), of which there is only one;
- The Data Server (DS), of which there can be many;
- The Data Client (DC), of which there are many;
- The Original Equipment Manufacturer (OEM) of which there are many;
- Domain Coordinators, of which there may be many.

A more detailed explanation of these terms is given below. Details of the roles for each of the scheme participants are managed by the IHO acting as the Scheme Administrator.

15-4.1 Scheme Administrator

The Scheme Administrator (SA) is solely responsible for maintaining and coordinating the Protection Scheme. The SA role is operated by The International Hydrographic Organization (IHO) on behalf of the IHO Member States and other organizations participating in the Protection Scheme. These organizations can have a coordinating role for a maritime product domain; for example IMO and IALA. The IHO as the SA will establish procedures with product domain operators using the Protection Scheme to protect their products. These procedures will enable these domain coordinators to digitally sign the digital certificates used by their member organizations to participate in the Protection Scheme.

The SA is responsible for controlling membership of the Scheme and ensuring that all participants operate according to defined procedures. The SA maintains the top level digital root certificate used to operate the Protection Scheme and which forms the root identity in the authentication chain.

The SA is responsible for distributing the manufacturer ID (M_ID) and manufacturer key (M_KEY) directly to all registered Data Servers participating in the Protection Scheme.

The SA is also the custodian of all documentation relating to the implementation of this part of S-100. All operational procedures are defined and managed by the SA.

15-4.2 Data Servers

Data Servers (DS) are responsible for the encryption and/or digital signing of the datasets in compliance with the procedures and processes defined in this Part. Data Servers may also issue Licenses (data permits) so that Data Clients, with valid user permits, can decrypt the product data.

Data Servers will use the M_KEY and M_ID information, as supplied by the SA, to issue encrypted product keys to each specific installation. Even though the keys used to encrypt each dataset are the same for individual data clients, they will be encrypted using the unique HW_ID and therefore cannot be transferred between other system installations from the same manufacturer.

The Scheme does not impede agents or distributors from providing data services to their customers. Agreements and structures to achieve this are outside the scope of this document. This document contains only the technical specifications to produce protected datasets compliant with this standard.

Hydrographic Offices, data producers, Value Added Resellers and RENC Organizations are examples of Data Servers.

15-4.3 Data Clients

Data Clients (DC) are the end users of datasets and will receive protected information from the Data Servers to access and use the datasets and services. The Data Client's software application (OEM System) is responsible for authenticating the digital signatures applied to the product files and decrypting the dataset information in compliance with the procedures defined in the Scheme.

Navigators with ECDIS/ECS systems are examples of Data Clients.

15-4.4 Original Equipment Manufacturers

Original Equipment Manufacturers (OEMs) subscribing to the S-100 Data Protection Scheme must build a software application according to the specifications set out in this document and self-verify and validate it according to the terms mandated by the SA. This Part will establish test data for the verification and validation of OEM applications for various S-100 based Product Specifications when products become available. The SA will provide successful OEM applicants with their own unique manufacturer key and identification (M_KEY and M_ID).

The manufacturer must provide a secure mechanism within their software systems for uniquely identifying each end user installation. The Scheme requires each installation to have a unique hardware identifier (HW_ID).

The software application will be able to decrypt the product keys in the data permits using the HW_ID stored in either the hard lock or soft lock devices attached to or programmed within the application to subsequently decrypt and uncompress the dataset files. Product integrity can be verified by authenticating the digital signature provided with the dataset files.

15-4.5 Domain Coordinator

Domain Coordinators are nominated, trusted bodies of the SA, able to produce certificates and provide intermediate authentication of Data Servers within their domain. Domain Coordinators are appointed by the SA and have delegated authority to sign Data Server certificates within their own domain. When Data Clients authenticate the identity of digital signatures created by Data Servers the certificates form a "chain" to the SA's root level identity. If a Data Server is certified by a Domain Coordinator, then the Data Client should also verify their identity against the SA root providing an authentication chain from the dataset to the SA.

15-4.6 Participant Relationships

The Scheme Administrator (SA), of which there can only be one, authenticates the identity of the other participants within the scheme. All Data Servers, Domain Coordinators and System Manufacturers

(OEMs) must apply to the SA to become participants in the Scheme and, on acceptance, are supplied with proprietary information unique to them. Data Clients are customers of Data Servers and OEMs, where Data Servers supply data services and OEMs the equipment to decrypt and display these services.

The SA will sign the public key of Data Servers to create their digital certificate to be used in the operation of the Protection Scheme. It is also possible for Domain Coordinators to sign the public key of their member organizations to create their digital certificates. The Domain Coordinators will inform the SA of each Data Server's identity and contact details and sign their certificates. The SA and Domain Coordinators will distribute M_ID and M_KEY information directly to all Data Servers participating in the Protection Scheme when they join the scheme and as more Data Clients are added.

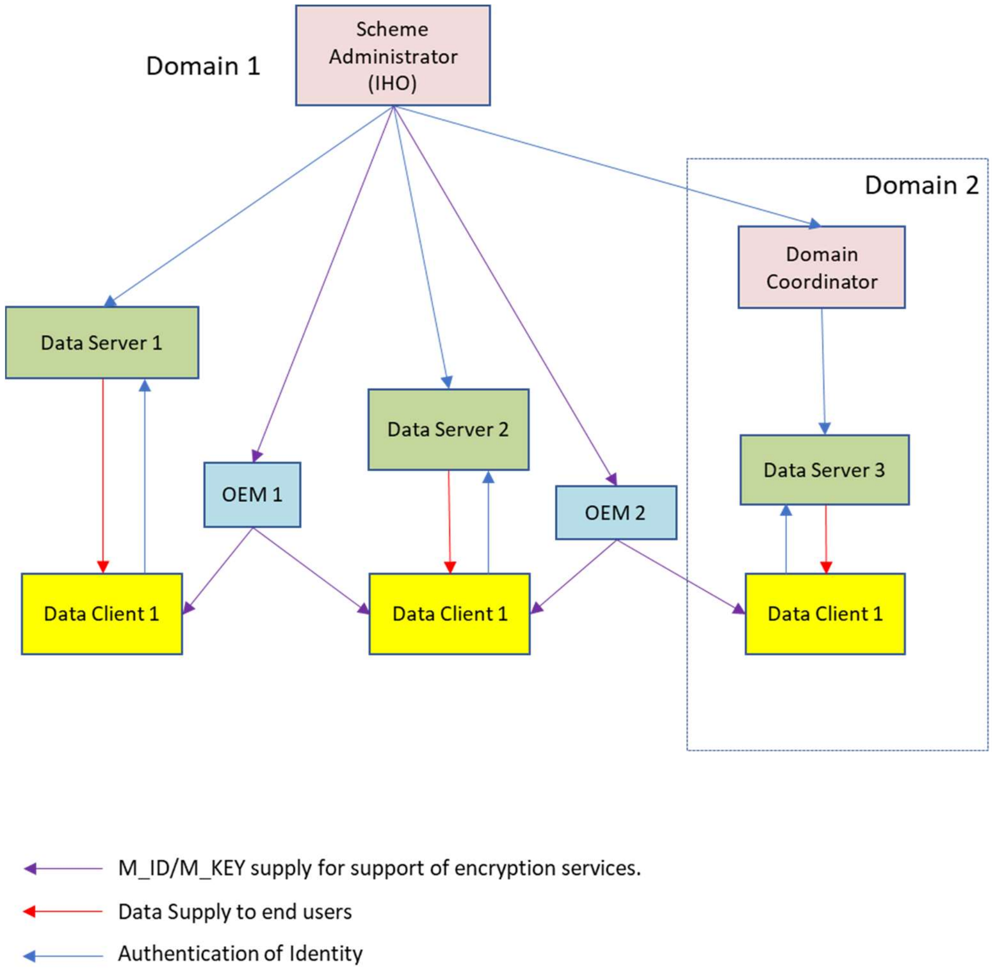


Figure 15-1 – Relationship between Protection Scheme participants

Since the Protection Scheme does not rely on Data Clients always having an internet connection to authenticate certificates or for certificate path validation, sufficient information shall be included in the Exchange Set Metadata to perform these functions. In all cases the SA certificate is installed on end user systems separately and not distributed in the exchange set metadata to provide independent verification of the SA certificate.

15-5 Data compression and packaging

15-5.1 Overview

The content of products based on the S-100 Data Model will, because of their structure, contain repeating patterns of information. Examples of this are small variations in the coordinate information within the file.

If compression is applied, the files are always compressed before they are encrypted as the effectiveness of any compression algorithm relies on the existence of structured data contents. The individual S-100 based Product Specifications will specify in metadata whether compression is being used.

All exchange set files must be digitally signed before any compression is applied.

15-5.2 Compression Algorithm

The Protection Scheme uses the ZIP algorithm to compress and uncompress files. The compression method is DEFLATE. Each file is compressed into a single file archive with the same name as the source file. If it is required to compress multiple files (for example, a Portrayal Catalogue) then they shall be located in a single root folder and the name of the compressed file set to the name of the root folder.

The encryption and digital signature features of ZIP are not used.

15-5.3 Encoding

The individual S-100 based Product Specifications will provide more details if compression is being used, and which files will be compressed.

The use of compression will be encoded:

- S-100_ExchangeCatalogue-compressionFlag with value **1**.

15-6 Data encryption

15-6.1 What Data is encrypted?

Any Product Specification that is based on the S-100 Data Model must define whether encryption will be used and which files will be encrypted.

When encrypted, the encryption algorithm must be the Advanced Encryption Standard (AES) in Cipher Block Chaining (CBC) mode of operation. It is always assumed that the complete file will be encrypted.

In addition the OEM System HW_ID (hardware ID) will be encrypted and provided to the Data Client in the form of a user permit. The keys used to encrypt the files are themselves encrypted by the Data Server and supplied to Data Clients as data permits. Information about the encryption algorithm is available in clause 15-6.2.1.

15-6.2 How is it encrypted?

Each single product is encrypted using a unique key. The same key is used to encrypt all files associated with the product and all updates issued for the product edition. The Scheme, however, allows for the keys to be changed at the discretion of the Data Server. The keys are delivered to Data Clients in the form of data permits.

15-6.2.1 Encryption algorithm

For encryption of permits and data files the Advanced Encryption Standard (AES) block cipher algorithm is used. This is a symmetric-key algorithm. This means that the same key is used for encryption and decryption. The algorithm defines how one block of plain text is converted to one block of cipher text and vice versa. The block size of the AES is always 16 Bytes (128 bit). The key length can be chosen from 128 bit, 192 bit or 256 bit. The corresponding variants are named AES-128, AES-192, or AES-256. In this Part of S-100 a 128 bit key length is always used.

The AES algorithm can only encrypt one block of plain text. For larger messages a block cipher mode of operation shall be used. This Protection Scheme chooses the Cipher Block Chaining (CBC) mode for encryption of more than one block of data. In this mode of operation it is required that the length of the plain text must be an exact multiple of the block size; padding is required.

15-6.2.2 Encryption padding

The padding methods that will be used is described in PKCS#7. It adds N bytes to the message until its length is a multiple of 16 Bytes. The value of each byte is N. Note that if the original plain text has already a multiple of 16 as length a full block of 16 bytes each having the value of 16 must be added.

Table 15-1 – Plain text padding

Plain text	Padded plain text
xx	xx **0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F**
xx xx	xx xx **0E 0E 0E 0E 0E 0E 0E 0E 0E 0E 0E 0E 0E 0E**
xx xx xx	xx xx xx **0D 0D 0D 0D 0D 0D 0D 0D 0D 0D 0D 0D 0D**
xx xx xx xx	xx xx xx xx **0C 0C 0C 0C 0C 0C 0C 0C 0C 0C 0C 0C**
xx xx xx xx xx	xx xx xx xx xx **0B 0B 0B 0B 0B 0B 0B 0B 0B 0B 0B**
xx xx xx xx xx xx	xx xx xx xx xx xx **0A 0A 0A 0A 0A 0A 0A 0A 0A 0A**
xx xx xx xx xx xx xx	xx xx xx xx xx xx xx **09 09 09 09 09 09 09 09 09**
xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx **08 08 08 08 08 08 08 08**
xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx **07 07 07 07 07 07 07**
xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx **06 06 06 06 06 06**
xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx **05 05 05 05 05**
xx xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx xx **04 04 04 04**
xx xx xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx xx xx **03 03 03**
xx xx xx xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx xx xx xx **02 02**
xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx **01**
xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx **10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10**

xx = Arbitrary Bytes

15-6.2.3 AES encryption CBC mode

In CBC mode each block of plain text is XORed with the previous cipher text block before being encrypted. An initialization vector IV is required for the first block. The mathematical formula is:

$$C_i = E_K(P_i \oplus C_{i-1});\ i \geq 1 \qquad (3a)$$
$$C_0 = IV \qquad (3b)$$

C_i is the i[th] block of cipher text; P_i is the i[th] block of plain text. E_K is the encryption method of AES encrypting exactly one block. IV is the initialization vector, and $\oplus$ is the XOR operation.

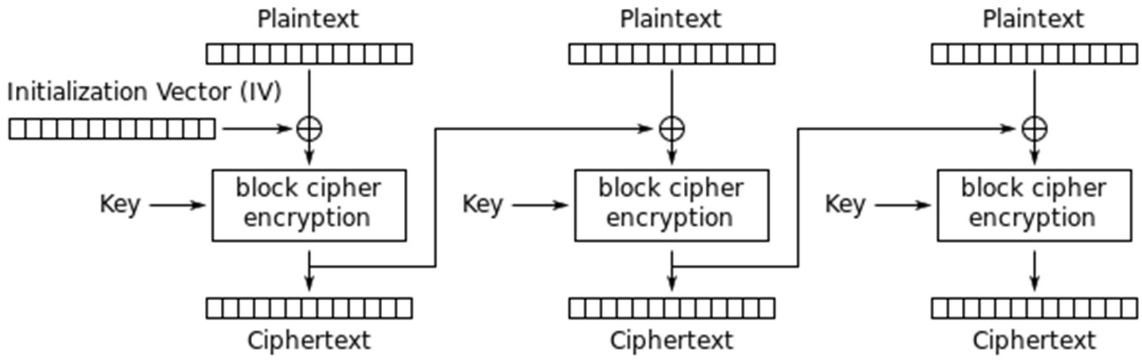


Figure 15-2 – Cipher Block Chaining (CBC) mode encryption (Source: Wikipedia)

Decryption is defined as:

$$P_i = D_K(C_i) \oplus C_{i-1};\ i \geq 1 \tag{4a}$$
$$C_0 = IV \tag{4b}$$

D_K is the decryption method of AES decrypting exactly one block.

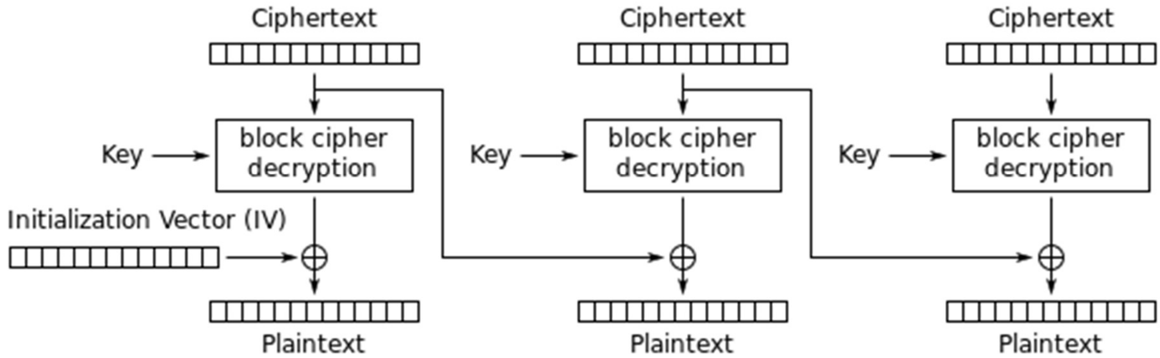


Figure 15-3 – Cipher Block Chaining (CBC) mode decryption (Source: Wikipedia)

15-6.2.4 AES CBC mode – initialization vector

Normally the initialization vector must be transferred from the encryption to the decryption. However an incorrect IV at the decryption will only corrupt the first plain text block. This can be easily recognized from the formulas and the diagrams. Each plain text block depends only on two adjacent cipher text blocks.

This behaviour will be used in the following modification of the CBC mode.

On encryption of data files the plain text will be prepended by a single random block. Then encryption is done as normal using a random initialization vector. This vector does not have to be transferred to the decryption at the Data Client.

On decryption an arbitrary initialization vector can be used and after normal CBC decryption the first plain text block is discarded. The rest is the original plain text data file.

This procedure does not require the transport of the IV or the use of a predicted IV. The first option would complicate the process of data transfer and the second would make it vulnerable to attacks especially if the first blocks of plain text are commonly known (as ISO/IEC 8211 Data Descriptive Records).

15-6.2.5 AES examples

The following examples are taken from the FIPS documentation.

Encrypting and decrypting of exactly one block:

Key_{128}: K = {00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 0a, 0b, 0c, 0d, 0e, 0f}
Plain Text: P = {00, 11, 22, 33, 44, 55, 66, 77, 88, 99, aa, bb, cc, dd, ee, ff}
Cipher Text: C = {69, c4, e0, d8, 6a, 7b, 04, 30, d8, cd, b7, 80, 70, b4, c5, 5a}

Key_{192}: K = {00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 0a, 0b, 0c, 0d, 0e, 0f,
10, 11, 12, 13, 14, 15, 16, 17}
Plain Text: P = {00, 11, 22, 33, 44, 55, 66, 77, 88, 99, aa, bb, cc, dd, ee, ff}
Cipher Text: C = {dd, a9, 7c, a4, 86, 4c, df, e0, 6e, af, 70, a0, ec, 0d, 71, 91}

Key_{256}: K = {00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 0a, 0b, 0c, 0d, 0e, 0f,
10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1a, 1b, 1c, 1d, 1e, 1f}
Plain Text: P = {00, 11, 22, 33, 44, 55, 66, 77, 88, 99, aa, bb, cc, dd, ee, ff}
Cipher Text: C = {8e, a2, b7, ca, 51, 67, 45, bf, ea, fc, 49, 90, 4b, 49, 60, 89}

The following example documents the modified CBC mode:

Key_{128}: K = {12, 34, 56, 78, 9a, bc, de, f0, 12, 34, 56, 78, 9a, bc, de, f0}
Plain Text: P = {fe, dc, ba, 98, 76, 54, 32, 10}
Plain Text after prepending a random block:
P’ = {48, d2, 4e, 7c, 00, 2f, 67, 4e, 93, 1d, ee, 27, 42, 17, a3, 4c}
{fe, dc, ba, 98, 76, 54, 32, 10}
Plain Text (padded):
P” = {48, d2, 4e, 7c, 00, 2f, 67, 4e, 93, 1d, ee, 27, 42, 17, a3, 4c}
{fe, dc, ba, 98, 76, 54, 32, 10, 08, 08, 08, 08, 08, 08, 08, 08}
Initialization vector (random):
IV_E = {45, b5, 00, d7, 28, 39, 42, bb, 85, 61, 28, d5, 97, 15, ca, 25}
Cipher Text using CBC Mode:
C = {ba, 45, ee, 06, 02, a6, 29, 35, 7a, e3, 90, 2c, 22, 4d, d9, d5}
{dd, 3b, 07, 3b, 84, 7f, 4d, 43, 28, 71, 19, 43, 97, d9, a6, 03}

For the decryption an arbitrary initialization vector can be used; for example:

IV_D = {00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00}

Decryption using the CBC will give the following plain text. The bytes added by the padding are already removed:

P_D‘ = {0d, 67, 4e, ab, 28, 16, 25, f5, 16, 7c, c6, f2, d5, 02, 69, 69}
{fe, dc, ba, 98, 76, 54, 32, 10}

Note that the first block is different from the one in P‘.

After discarding the first block the original message is recovered.

P_D = {fe, dc, ba, 98, 76, 54, 32, 10} = P

15-7 Data encryption and licensing

15-7.1 Introduction

Data Clients generally do not buy S-100 based products but are licensed to use them. Licensing is the method that Data Servers use to give Data Clients selective access to up-to-date products for a given period of time.

To operate the scheme effectively there must be a means where Data Client systems can unlock the encrypted data. To unlock the data the Data Clients system must have access to the keys that were used to encrypt the licensed data files. These keys are supplied to the Data Client, encrypted, in a permit file containing a set of permits. It is these data permits that contain the encryption keys. This method is used for file based exchange of data between the Data Client and Data Server. Other frameworks and methodologies, such as data streaming may use either variations of algorithms or different key lengths, specifying in metadata how they are defined.

To make each set of data permits exclusive the keys must be encrypted using something that is unique to the Data Clients system. OEMs assign a unique identifier (HW_ID) to each of their systems and provide an encrypted copy of this, in the form of a user permit, to each Data Client. The HW_ID is encrypted and stored in the user permit.

OEMs encrypt the HW_ID with their own unique manufacturer key (M_KEY) so that a HW_ID cannot be duplicated by another manufacturer. The IHO, as the Scheme Administrator, provides the Data Servers with access to the OEM M_KEYs and can therefore decrypt the HW_ID stored in the user permit. Data Servers encrypt their dataset keys with the manufacturers HW_ID when producing a set of data permits. This makes them unique to the Data Client and as such not transferable between Data Client systems.

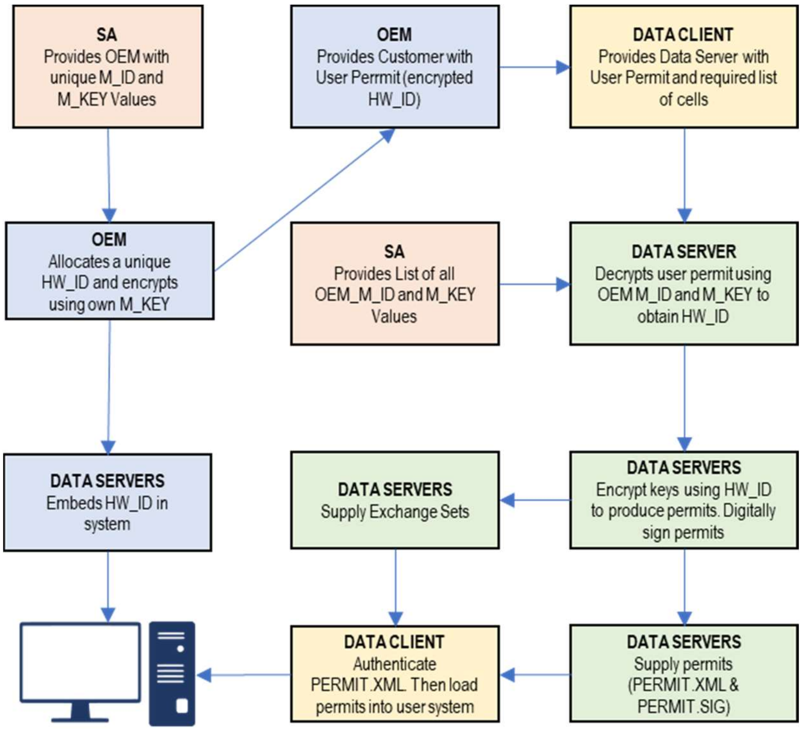


Figure 15-4 – High level licensing diagram based on S-101 ENC products

15-7.2 Conversion of bit strings to integers

15-7.2.1 Converting bit strings to an integer

A sequence of bits $\{b_1, b_2, \ldots, b_n\}$ defines an unsigned integer I number by:

$$I = b_1 2^{n-1} + b_2 2^{n-2} + \cdots + b_{n-1} 2^1 + b_n;\ b_i \in \{0,1\} \quad (1a)$$

Or

$$I = \sum_{i=1}^{n} b_i 2^{n-i} \quad (1b)$$

The bit b_1 is the most significant bit and the bit b_n is the least significant bit of the sequence. The integer will be in the range: $0 \leq I < 2^n$.

In most implementations the bit string will be organized as a sequence of bytes $\{B_0, B_1, \ldots, B_m\}$, with:

$B_{m-j} = \{x_{n-8j-7}, x_{n-8j-6}, \ldots, x_{n-8j}\}; \ \forall j \in \{0 \ldots m\}$ with $x_i = \{b_i; \ \forall i \ i > 0 \ 0; \ \forall i \ i \leq 0$ and $m = \lceil \frac{n}{8} \rceil$ (2)

A possible implementation of converting such a byte sequence to an integer number is given by the following pseudo code.

Input: Byte sequence *B={B₀, B₁,…,Bₘ}*

Output: non-negative integer number *I*

Let I=0
for k from 0 to m
$I = I * 2^8$
$I = I + B_k$
Return I

15-7.2.2 Converting an integer number to a bit string

Formula 1a and 1b describe how a bit string is related to a corresponding (non-negative) integer number. Assuming that the bit string is organized as a sequence of bytes as defined by (2) the following algorithm shows how to transform an unsigned integer number to a bit string.

Input: a non-negative integer number I with $0<=I<2^n$
Output: a sequence of bytes B of length $m = \{1; I = 0 \ \lceil \frac{n}{8} \rceil; I > 0$
Let B be an empty sequence
If I = 0
Append the byte b=0 to B
Else
While I > 0 do
Let $c = I \ mod \ 2^8$
Prepend c to B
Let $I = I \ div \ 2^8$
While the length of B is < m
Prepend 0 to B
Return B

Note that the division by 2^8 is equivalent by the bit shift operation $I>>8$

15-7.2.3 Converting an unsigned integer number to a hexadecimal text representation

The following pseudo code shows how to convert an unsigned integer number to its hexadecimal text representation. In this text representation each digit can have 16 different values.

The integer *I* is defined as:

$$I = d_n 16^{n-1} + d_{n-1} 16^{n-2} + \cdots + d_2 16 + d_1 \quad (3)$$

Table 15-2 – Conversion of unsigned integer to hexadecimal text

Digit d	Bit string	Character	ASCII Code (Hex)	ASCII Code (dec)
0	0000	‘0’	30	48
1	0001	‘1’	31	49
2	0010	‘2’	32	50
3	0011	‘3’	33	51
4	0100	‘4’	34	52
5	0101	‘5’	35	53
6	0110	‘6’	36	54
7	0111	‘7’	37	55
8	1000	‘8’	38	56
9	1001	‘9’	39	57

10	1010	'A'	41	65
11	1011	'B'	42	66
12	1100	'C'	43	67
13	1101	'D'	44	68
14	1110	'E'	45	69
15	1111	'F'	46	70

The algorithm is:

Input: An unsigned integer number I
Output: The hexadecimal text representation S
Let S be an empty sequence of characters.
If I = 0
    Let S = "0"
Else
    While I>0
        Let c be the character corresponding to the value $d = I \; mod \; 16$
        Prepend c to S
        Let $I \; = \; I \; div \; 16$
Return S

15-7.2.4 Converting a hexadecimal text representation to an unsigned integer number

The following algorithm shows how to convert a hexadecimal text representation of an unsigned integer number to the integer number itself.

Input: A hexadecimal text representation S of an unsigned integer number S = $\{s_1, s_2, \ldots, s_m\}$
Output: An unsigned integer number I
Let I = 0
For I = 1 to m
    *I = I*16*
    I = I + d; where d is the digit value corresponding to the character S_i
Return I

15-7.3 The User Permit

The user permit is created by OEMs and supplied to Data Clients as part of their system so that they can obtain the necessary access to encrypted products from Data Servers. The following section defines the composition and format of the user permit.

All Data Clients with systems capable of using data, protected in accordance with the IHO Data Protection Scheme, must have a unique hardware identification (HW_ID) defined by the data client built into their end-user system. Such a HW_ID is often implemented as a dongle or by other means ensuring a unique and tamperproof identification for each installation.

The HW_ID is unknown to the Data Client, but the OEM will provide a user permit that is an encrypted version of the HW_ID and unique to the Data Client's system. The user permit is created by taking the assigned HW_ID and encrypting it with the manufacturer key (M_KEY). The CRC32 algorithm is run on the encrypted HW_ID and the result appended to it. Finally the manufacturer attaches their assigned manufacturer identifier (M_ID) to the end of the resultant string. The M_KEY and M_ID values are supplied by the SA and are unique to each manufacturer providing IHO Data Protection Scheme compliant systems.

The Data Client gains access to S-100 based encrypted products by supplying their user permit to the Data Server. This enables the Data Server to issue Data Permits specific to the Data Client's user permit. Since the user permit contains the manufacturers unique M_ID this can be used by Data Servers to identify which M_KEY to use to decrypt the hardware ID in the user permit. The M_ID is the last six characters of the user permit. A list of the manufacturer M_KEY and M_ID values is issued and updated by the SA to all Data Servers subscribing to the scheme. This list will be updated periodically as new OEMs join the scheme.

15-7.3.1 Definition of user permit

The user permit is 46 characters long and must be written as ASCII text with the following mandatory encoding format and field lengths:

Table 15-3 – User permit field structure

Encrypted HW_ID	Check SUM (CRC)	M_ID Manufacturer ID
128 bits (32 characters)	8 characters	6 characters

Any alphabetic character will be written in upper case.

Example: Encoded user permit:

`AD1DAD797C966EC9F6A55B66ED98281599B3C7B1859868`

The structure of the user permit is explained in the following sub-clauses.

15-7.3.1.1 HW_ID Format

The HW_ID is a 16 byte hexadecimal number defined by the OEM. Such a HW_ID can be implemented as a dongle or by other means and must ensure a tamperproof identification of each installation.

The HW_ID will be stored in an encrypted form in the user permit. It is encrypted using the AES algorithm with the OEM M_KEY as the key resulting in a 128 bit value (see clause 15-6.2.4). The 128 bit encrypted HW_ID is then represented in its ASCII form in the user permit as 32 hexadecimal digits, if necessary prepending 0's to get the 32 required digits.

Note that the size of the HW_ID is identical to the AES block size and does not require any padding.

Example of HW_ID is: `40384B45B54596201114FE9904220101`

Example of encrypted HW_ID is: `AD1DAD797C966EC9F6A55B66ED982815`

(M_KEY=`4D5A79677065774A7343705272664F72`)

15-7.3.1.2 Check Sum (CRC) Format

The Check Sum is an 8 digit hexadecimal number. It is generated by taking the encrypted HW_ID and converting it to a 32 character hexadecimal string. The string is then hashed using the algorithm CRC32 and the 4 bytes converted to an 8 character hexadecimal string.

The Check Sum is not encrypted and allows the integrity of the user permit to be checked.

The Check Sum in the above example is calculated from:

- Example HW_ID: `40384B45B54596201114FE9904220101`
- Example Encrypted HW_ID: `AD1DAD797C966EC9F6A55B66ED982815`
- CRC32 Checksum: `99B3C7B1`

15-7.3.1.3 M_ID Format

The M_ID is a 6-character alphanumeric code expressed as ASCII text provided by the SA to the OEM. The SA will provide all licensed manufacturers with their own unique Manufacturer Key and Identifier (M_KEY and M_ID) combination. The manufacturer must safeguard this information.

The SA will provide all licensed Data Servers with a full listing of all manufacturer codes as and when new manufacturers subscribe to the scheme. This information is used by the Data Server to determine which key (M_KEY) to use to decrypt the HW_ID in the User permit during the creation of Data Client Dataset Permits.

The M_ID in the above example is: `859868`

15-7.3.2 M_KEY Format

The M_KEY is a random 16 byte hexadecimal (128 bit) number assigned to the manufacturer and provided by the SA. The OEM uses this key to encrypt assigned HW_ID values to generate user permits. This key is also used by the Data Server to decrypt assigned HW_IDs. Note that the size of the M_KEY is identical to the AES block size and does not require any padding.

Example of the M_KEY is: 4D5A79677065774A7343705272664F72 (Hexadecimal representation)

The complete example is shown in Table15-4 below:

Table 15-4 – Complete user permit – example

Field	Value
M_ID	859868
M_KEY	4D5A79677065774A7343705272664F72
HW_ID	40384B45B54596201114FE9904220101
Encrypted HW_ID	AD1DAD797C966EC9F6A55B66ED982815
CRC32 (Encrypted HW_ID)	99B3C7B1
Complete User Permit	AD1DAD797C966EC9F6A55B66ED98281599B3C7B1859868

15-7.4 The data permit

To decrypt a data file the Data Client must have access to the encryption key (see clause 15-6.2.1) used to encrypt it. Since the encryption keys are only known to the Data Server there needs to be a means of delivering this information to Data Clients in a protected manner. This information is supplied by the Data Server to the Data Client in an encrypted form known as a permit. A file is provided to deliver the data permit and it is named PERMIT.XML (see clause 15-7.4.1). This file may contain several permits based on the product coverage required by the Data Client.

The PERMIT.XML file will be delivered either on hard media or using online services in accordance with the Data Servers operating procedures. These procedures will be made available to Data Clients when purchasing a license.

Each record within the data permit file also contains additional fields that are supplied to assist OEM systems to manage the Data Clients license and permit files from multiple Data Servers, see clause 15-7.4.2.

Data Clients can obtain a licence to access products by supplying the Data Server with their unique user permit (see clause 15-7.3). Data Servers can then extract the HW_ID from the user permit, using the Data Client's M_KEY, and create client specific permits based on this value. The format of a permit file record is described below in clauses 15-7.4.1 to 15-7.4.4.

Since data permits are issued for a specific HW_ID they are not transferable between installations (Data Client Systems). This method of linking the permit to the installation supports the production of generically encrypted data which can be distributed to all Data Clients subscribing to a service.

The Data Clients system decrypts the permit using the assigned HW_ID stored by hardware or software means. The decrypted keys can then be used by the system to decrypt the licensed products. Since several Data Servers can make permit files for a specific type of product, it is the responsibility of the Data Client system to manage permit files from multiple Data Servers.

15-7.4.1 The permit file (PERMIT.XML)

The filename will always be provided in UPPERCASE as will any alphabetic characters contained in the file. The file is completely encoded in ASCII and conforms to the S-100 XML schema for permits. OEMs should be aware that all ASCII text files generated by the Protection Scheme may contain ambiguous end-of-line markers such as CR or CRLF and should be able to deal with these.

The XML schema structure is illustrated in Figure 15-5 below.

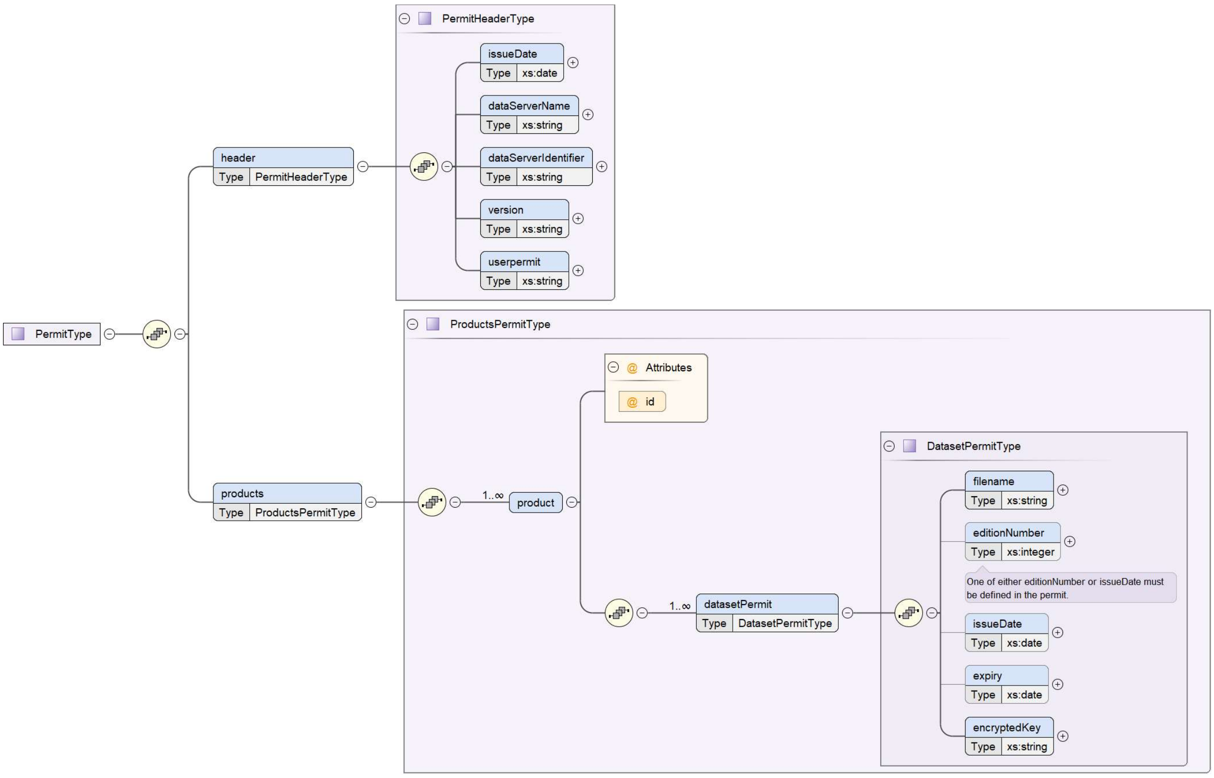


Figure 15-5 – Structure of the permit file

The PERMIT.XML file can contain multiple sections with a corresponding XML element as follows:

Table 15-5 – PERMIT.XML elements

XML element	Description
header	File creation date, the name of the Data Server and the format version
products	Permits from the Data Server for the specified product

Note that the PERMIT.XML file can contain permits for multiple products provided by the Data Server. OEMs must ensure that their end-user software is able to merge permits from multiple data servers.

15-7.4.2 The Permit File - Header content

The following Table defines the content and format of each section within the permit XML file.

Table 15-6 – Contents and format of PERMIT.XML

Content	XML element	Description
File name	filename	Name of resource the permit is intended for, without pathname Format: Character string
Date and time	issueDate	Date and time XML format: xs:dateTime Example: <issueDate>20180320T17:11:00Z</issueDate>
Provider	dataserverName	Name of Data Server who has generated the permit file. The Data Server name should be consistent and use the same organizational contact as defined in S100_ExchangeCatalogue – contact XML format: xs:string

Provider identifier	dataserverIdentifier	Short identifier of data server
Version	version	Version number of S-100. It will be compatible with the IHO version numbering scheme X.Y.Z. For example 4.0.0 Format: Character string
User permit	userpermit	The user permit that the permit is intended for. This allows the client system or implementer to validate the destination. The end-user system must be capable of checking if the permit is for the designated system on a multi system bridge. Character string as defined in clause 15-7.3.1 Format: Character string

15-7.4.3 Product sections and permit records fields

The header element in the PERMIT.XML file is followed by a single element called “products” which contains multiple “product” records, each of which contain the actual permits for those products. This allows a single PERMIT.XML file to contain permits for multiple products all destined for a single end user system. The attribute “id” for each product section contains the S-100 identifier of the Product Specification to which the permits relate; for example, <product id=”S-101”>.

15-7.4.4 Definition of the permit record

Each product element in the PERMIT.XML file contains a sequence of “permit” elements. These elements contain the actual permits for the products identified. The Table below defines the elements contained in the permit elements with a definition of the purpose of each; fields are mandatory unless otherwise stated. Note that permits are only issued for Base datasets and the same permit is used to decrypt incremental updates (if the Product Specification implements updates).

Table 15-7 – Permit record elements

Field	Purpose	Format
filename	The file name as defined in S100_DatasetDiscoveryMetadata – fileName. It enables Data Client systems to link the correct encryption key to the corresponding encrypted file. The pathName to the file is defined in the Exchange Set Metadata	Character string
editionNumber	[Optional] The edition number of the product file as defined in S100_DatasetDiscoveryMetadata - editionNumber For products without an edition number the permit will apply to all issued datasets	Character string
issueDate	[Optional] If the product does not have an edition number then the issue date may be used as an alternative identifier	xs:date
expiry	This is the date when the Data Clients licence expires. Systems must prevent any new editions or updates issued after this date from being installed	xs:date

encryptedKey (EK)	EK contains the decryption key for the specified edition of the product file	32 character hexadecimal string representing the 128 bit encrypted key

15-7.4.5 Permit file signatures

Each permit file will have a digital signature created by the Data Server. The digital signature will be stored in a separate file and will reuse the name of the permit file but will have ".SIGN" appended, for example permit.sign.

The content of the signature file will be the Data Server certificate and the permit file signature and it shall be encoded in accordance with the S-100 XML Schemas. The OEM system shall authenticate the Data Server certificate before authenticating the permit file before the dataset permit keys are decrypted.

15-7.4.6 An example PERMIT.XML file

```
<?xml version="1.0" encoding="utf-8"?><permit
xmlns="http://www.iho.int/s100/se">
    <header>
        <issueDate>20180320T17:11:00Z</issueDate>
        <dataserverName>Primar</dataserver>
        <dataserverIdentifier>PR</dataserver>
        <version>1.0.0</version>
        <userpermit>
            267C3AD506E69B1ED18AA5ECC7FFDE6E7C330CE8859868
        </userpermit>
    </header>
    <products>
        <product id="S-101">
            <datasetPermit>
                <filename>101GB40079ABCDEF.000</filename>
                <editionNumber>10</editionNumber>
                <expiry>20223112</expiry>
                <encryptedKey>
                       2E16E07E451FF1854156634DA3DD3FB8
                </encryptedKey>
            </datasetPermit>
            <datasetPermit>
                <filename>101NO32802411223.000</filename>
                <editionNumber>5</editionNumber>
                <expiry>20220610</expiry>
                <encryptedKey>
                       C714B5C0FBDF14BFE4B1F12E62CE5FF6
                </encryptedKey>
            </datasetPermit>
        </product>
        <product id="S-102">
            <datasetPermit>
                <filename>102NO329048208.h5</filename>
                <editionNumber>1</editionNumber>
                <expiry>20221231</expiry>
               <encryptedKey>
                      50BBC28B6793E1C3966B45FB2932E1BE
               </encryptedKey>
            </datasetPermit>
        </product>
    </products>
</permit>
```

15-8 Data authentication

This section specifies the mechanisms, structures and content required for the implementation of copy protections and/or authentication methods by S-100 Product Specifications. It defines standardized methods for the encryption of file based components of datasets as well as Feature and Portrayal Catalogues. Algorithms and methods for digital signature implementation are defined as well as the surrounding infrastructure required for key management and identity assurance within the IHO Data Protection Scheme.

15-8.1 Introduction to data authentication and integrity checking

The digital signature technique in S-100 uses a standard algorithm and key exchange mechanism widely available and used. Digital signatures use asymmetric public key algorithms within a PKI-like infrastructure scheme to unbreakably bind a data file with the identity of the issuer.

The Scheme relies on asymmetric encryption[1] of a checksum of a data file. By verifying the signature against the issuer's public key, and also verifying the issuer's public key against a top level identity, the user is assured of the signer's identity. A detailed technical description of digital signatures is beyond the scope of this document and the reader is referred to the Digital Signature Standard (DSS – FIPS Publication 186) for a more detailed and accessible explanation. This Part of S-100 assumes a basic knowledge of digital signature terms and the operation of PKCS (public key cryptography standards) authentication schemes.

The IHO Data Protection Scheme can be considered to have three distinct phases:

1) A Scheme Administrator (SA) verifies the identity of a Data Server of S-100 products and provides the supplier with information to allow them to digitally sign their products.
2) A Data Server issues products signed with their identity (and their identity's verification by the SA).
3) The subsequent verification by the Data Client of the Data Server's identity; its association with the SA; and the integrity of the product data.

A Domain Coordinator may also act as an intermediary between the Data Server and the SA. The SA certifies the identity of the Domain Coordinator who then, in turn, can certify the identities of Data Servers they are responsible for.

It should be noted that the S-100 digital signature mechanism is not intended solely for S-100 Product Specifications' data files. It is possible to both encrypt (and issue permits for) and digitally sign any file based data and the mechanisms described in this Part will be used to sign catalogues and other supplementary files, including Feature and Portrayal Catalogues.

[1] Asymmetric cryptography relies on algorithms where encryption and decryption take place with different cryptographic keys. Therefore one person can encrypt data and make available a decryption key for others to decrypt it. These keys are referred to as the "private key" and the "public key", collectively known as a "key pair".

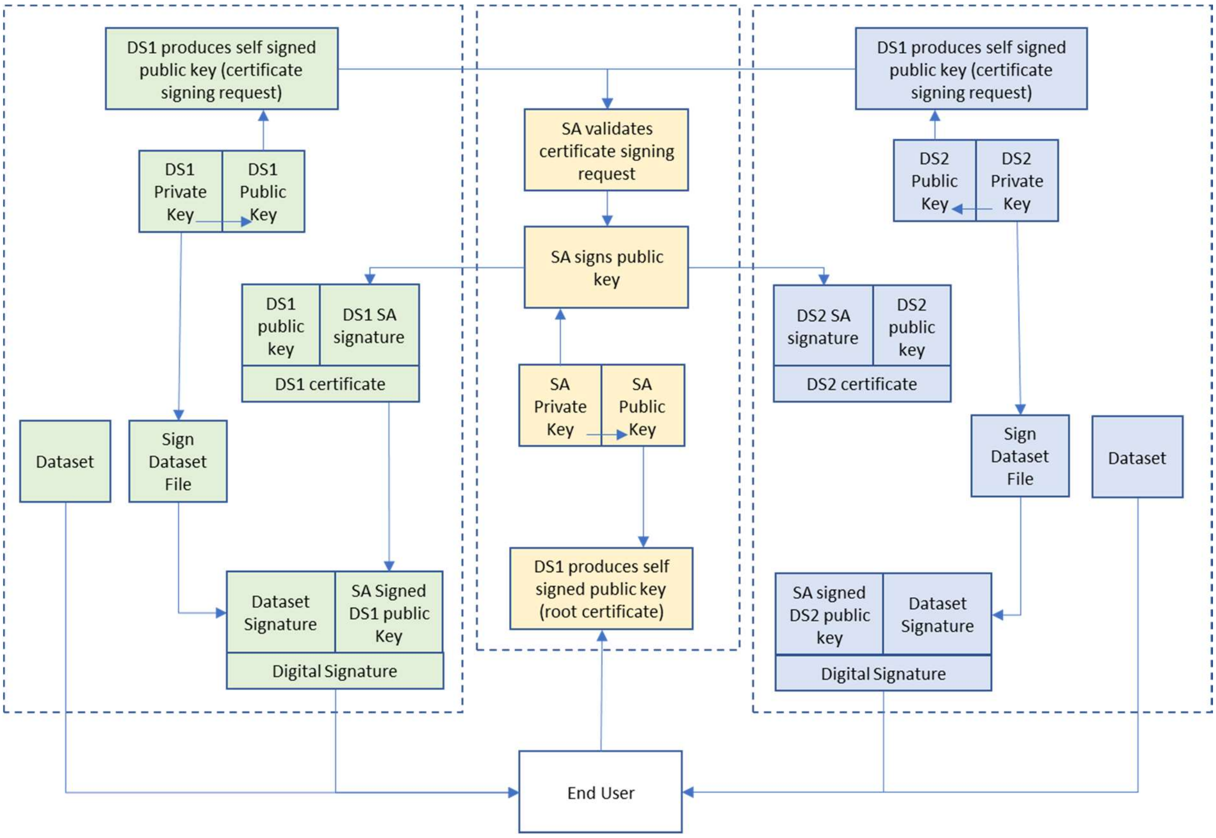


Figure 15-6 – The process of data server and digital signature creation

15-8.2 Data Protection Scheme setup, Data Server signup and authentication sequence

The following is a list of the steps taken by each body in the Data Protection Scheme during the digital signing of data files.

1. Scheme Creation and Setup (once only, at the instigation of the Data Protection Scheme):
   a. The SA creates their own public/private key pair and self-signs it.
   b. The SA puts their self-signed Public Key (also known as their “certificate”) in the public domain.
   c. The SA Public Key is embedded where required in OEM systems.
2. Data Server setup (once only):
   a. The Data Server creates a Public and Private Key pair.
   b. The Data Server signs their Public Key (with their Private Key) creating a Self Signed Key (also sometimes called a “certificate signing request”).
   c. The Data Server’s Self Signed Key (SSK) is sent to the SA (or Domain Coordinator) for validation when applying to join the IHO S-100 Data Protection Scheme. Any other requirements and duties within the Data Protection Scheme are issued to the prospective Data Server at this stage.
3. Data Server Identity Verification:
   a. If accepted the SA verifies the Data Server’s SSK and identity.
   b. The SA signs the Data Server’s SSK with its own Private Key to produce an SA signed Data Server Certificate.
   c. The Data Server certificate is then returned to the Data Server.
   d. The Data Server verifies that the certificate signs their Public Key against the SA Public Key.

4. The Data Server can then produce digital signatures of data files. Digital signatures of Feature and Portrayal Catalogues can also be produced by some Scheme participants as required.

15-8.3 Verification of digital signatures

The verification of digital signatures by a client system takes the following steps:

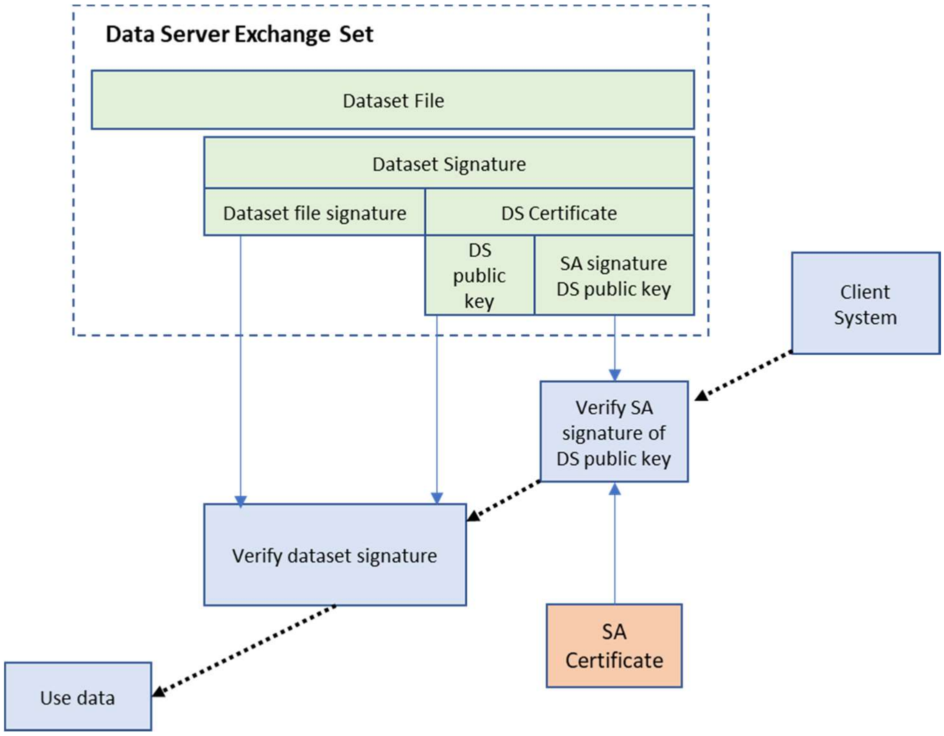


Figure 15-7 – The process of data authentication by a client system

15-8.4 Data Formats and standards for digital signatures, keys and certificates

The following categories of content are required for data authentication:

1. Key pairs, Private and Public Keys. These are all PEM encoded DSA keys together with their DSA key parameters. These keys should all be 2048 bits long.
2. Certificate signing requests and digitally signed Public Keys. When a Public Key is itself digitally signed it is referred to as a "certificate" (because the Public Key is "certified" by the use of the Private Key to authenticate it). When the Public Key is signed by its corresponding Private Key it is referred to as a "self-signed" certificate. These are laid out as X.509 records and can be either DER or PEM encoded to be sent to the SA for signing. When embedded within XML files keys should be PEM encoded so that the plain text can be inserted as an XML element.
3. The digital format of the SA signed) Public Keys ("certificates") is X509v3 format encoded as PEM.

The distinguished name (DN) in the X.509 certificate forms part of the immutable content of the certificate (that is, it cannot be changed without invalidating the certificate). The roles of the scheme participants and the domains they are assigned to may be encoded in the DN. The IHOs operational procedures for the Data Protection Scheme will implement whatever specific procedures are required for the formatting of this content. The SA may place restrictions on the values allowed in the DN's components (for example, the Common Name or the Organization) and the format of such identifiers in order to manage the operation of the Data Protection Scheme amongst its participants.

The policies and procedures implemented by the SA are not within scope of this Part of S-100 and shall be defined elsewhere. Using the DN to define the certificateRef fields can also assist implementers in selecting the correct certificate when verifying a digital signature. This may also be mandated by the SA as it specifies how the Data Protection Scheme is operated.

PEM format defines a textual encoding of the multiple large numbers required by the DSA algorithm (along with the DSA parameters required by the DSA algorithm). PEM encoding (originally developed for email encoding but used extensively in the encryption community for encoding of long integers used for keys and digital signatures) allows the embedding of Public Keys and Data Server certificates within XML files for permit file XML creation, the creation of catalogue and support file metadata and the production of digital signatures of Portrayal and Feature Catalogues. Digital signatures of S-100 data files must be embedded in the catalogue metadata and serve the dual purpose of a checksum against the unencrypted data file and the authentication of its source. Therefore they must be produced prior to any compression and encryption mechanism as copy protection is itself optional.

The SA Certificate represents a DSA Public Key of length 2048 bits provided as a PEM encoded text file. The S-100 Part 15 SA Certificate will always be available in a file called IHO.PEM. The IHO.PEM file is available from IHO at http://www.iho.int.

Digital Signatures in S-100 are implementations of the Digital Signature Standard (DSS). The DSS uses the Secure Hash Algorithm (SHA256) to create a message digest (hash) of the file content that are 256 bits long. The message digest is then input to the Digital Signature Algorithm (DSA) to generate the digital signature for the message using an asymmetric encryption algorithm and the 'Private Key' of the signer's key pair. S-100 file based authentication uses a DSA key length of 1024 bits. Other frameworks or data streaming via APIs may use different key lengths with interoperable formats.

In the DSA algorithm a signature is a sequence of two integers. By convention these are referred to as R and S (an "R,S pair"). The format of digital signatures when embedded in XML files is as follows:

```
<digitalSignature id="primar" certificateRef="root">
MEQCIEW75Fe5latHslBU8oHYmILBqKWv7taRQ0z1VS+FAJXCAiAmLtAZcXJUp9t6yuNrVSk2TWLN
Cn6UvuAW8guohfPKpg==</digitalSignature>
```

The encoding of the two R,S large integers is a Base64 ASN.1 byte sequence[2]. These are produced natively by the openssl implementation and can be generated and verified without the need to unpack the individual R and S integers. This encoding conveniently wraps the two values unambiguously into a byte array. The ASN.1 sequence representing the R,S pair is then Base64 (RFC 4648) encoded for representation in the XML digital signature elements.

The ASN.1 schema for the above example is:

```
SEQUENCE (2 elem)
  INTEGER (158 bit) 7B980FF65B48DF1D9A9396F918E37FC7B6B8F5D4
  INTEGER (158 bit) 73740AF5AA63116E23E57352B5B88D143BFC630C
```

The digital signature also contains the following attributes:

1. An "id" attribute to act as an identifier.
2. A certificateRef attribute identifying the dataserver certificate with the correct public key in it for authentication. If the signature is authenticated by the SA then the certificateRef is the identifier of the SA, defined in the schemeAdministrator element of the XML container type.

These attributes are described in Clause 15-8.8.

15-8.5 Creation of key material and certificate signing requests (signed Public Keys)

The commonly used "openssl package" provides a public domain, open source tool for production of key material in the required open standards specified within this Part.

[2] Abstract Syntax Notation One (ASN.1) is a standard interface description language for defining data structures that can be serialized and deserialized in a cross-platform way. It is broadly used in telecommunications and computer networking, and especially in cryptography. https://en.wikipedia.org/wiki/Abstract_Syntax_Notation_One

Table 15-8 below shows basic command line examples for creation of the Public and Private Key pairs, certificate production and digital signing of data files.

15-8.4.1 SA setup

This procedure is performed once only. The command SA-1 in the Table sets up a new set of DSA parameters and the SA-2 command creates the SA's "root certificate" - their self-signed key which self-certifies their identity.

When a Data Server creates an X509 certificate signing request (CSR), the SA signs it using command SA-3. This creates a SHA256 signed version of the Data Server's Public Key. The PEM encoded version of the "signedicds.crt" file is what is embedded in both permit files and catalogue metadata as the "Data Server certificate".

Table 15-8 – Creation of Public and Private Key pairs – basic commands

Task	Command
SA-1 create DSA parameters	`openssl dsaparam 2048 -out dsaparam.txt`
SA-2 create SA root key and self signed root certificate	`openssl req -x509 -sha256 -nodes -days 365 -newkey dsa:dsaparam.txt -keyout iho.key -out iho.crt`
SA-3 sign a verified certificate signing request	`openssl x509 -req -in  CSR.csr -sha256 -CA iho.crt -CAkey iho.key  -CAcreateserial -out signedicds.crt`

15-8.4.2 Data Server setup

The Data Server sets up their identity with the SA by using the once only process described by commands DS-1 to DS-5. This delivers an SA signed certificate to the Data Server which is included with every delivery of signed material to the Data Client.

Table 15-9 – Data Server setup commands

Task	Command
DS-1 Create DSA parameter file	`openssl dsaparam 2048 -out ICDSparam.txt`

DS-2 create a Data Server key
 	
DS-3 Split Public Key from Private Key
	`openssl req -out CSR.csr -new -newkey dsa:ICDSparam.txt -nodes -keyout icds.key`
	
	`openssl dsa -outform pem -in icds.key -out icdspubkey.txt -pubout`

DS-4 Create a certificate signing request	`openssl req -out CSR.csr -key icds.key -new`
DS-5 Verify received certificate from SA	`openssl verify -verbose -CAfile iho.crt signedicds.crt`

DS-6 Make data file
 	
DS-7 Sign data file
 	
DS-8 Create a hexadecimal version of the signature
 	
DS-9 Verify binary signature
	`echo "hello world" > hw.txt`
	
	`openssl dgst -sha256 -sign icds.key hw.txt > hw.sig`
	
	`xxd -u -ps hw.sig > data.txt`
	
	`(to convert back use xxd -r -u -ps data.txt > data.sig)`
	
	`openssl dgst -sha256 -verify icdspubkey.txt -keyform pem -signature hw.sig hw.txt`

The commands DS-6 to DS-9 show how a simple text file “hello world” can be created, signed with the Data Server’s private key to create a DSA-SHA256 signature, and then verified. DS-8 creates a hexadecimal format signature which can be translated into the following XML (including conversion to Base64) for embedding in an XML file (either PERMIT.XML or the catalogue metadata as required) according to the relevant Part of S-100.


```xml
<digitalSignature>
      MEQCIHVvkGrJl0joEqmS5PCmnJW4pydisZW5gpJGoU3CUeOVAiAZvuRA0y3QDLgnzJ8Il
      4oFX4U40BJ36UhRBVLUFfiVwQ==
</digitalSignature>
```


15-8.6 Digital certificate example

Digital certificates will be PEM encoded for easy exchange and embedding in XML files. The following is an example of a PEM encoded Data Server certificate. The commands listed in the previous section format the public keys and the certificate signing request appropriately for communication between the SA and the DS. When embedding the digital certificates in XML elements, the header and footer lines are omitted.

The catalogue file of a S-100 based Exchange Set will contain a copy of all the Data Server certificates in use by all the files included in the Exchange Set with the exception of the SA root certificate which is installed separately by the end user. An identifier representing the SA root certificate is included in the exchange catalogue certificates by a “schemeAdministrator” element with an “id” attribute.

Each XML element containing a certificate will have a unique identifier attribute “id”. Each XML certificate definition will also include an attribute, “issuer” defining the id of the issuer, either the SA (identified by the schemeAdministrator id) or a domain coordinator (whose certificate will also be included in the Exchange Set). The example below shows an extract from an exchange set catalogue header with the SA certificate (not included) given the id “root”. An SA signed Data Server certificate with id “DS1” is then included with the PEM encoded certificate.


```xml
    <S100XC:certificates>
        <S100CE:schemeAdministrator id="root"/>
        <S100CE:certificate id="DS1" issuer="root">
MIIExzCCBGwCCQDvlJSe+eVonDALBglghkgBZQMEAwIwgYIxCzAJBgNVBAYTAk1D
MQ8wDQYDVQQIDAZNb25hY28xMDAuBgNVBAoMJ0ludGVybmF0aW9uYWwgSHlkcm9n
cmFwaGljIE9yZ2FuaXphdGlvbjEwMC4GA1UEAwwnSUhPIERhdGEgUFJvdGVjdGlv
biBTY2hlbWUgQWRtaW5zdHJhdG9yMB4XDTIxMTIwNDA3MjkzOVoXDTIyMDEwMzA3
MjkzOVowYjELMAkGA1UEBhMCVUsxEzARBgNVBAgMClNvbWUtU3RhdGUxITAfBgNV
BAoMGEludGVybmV0IFdpZGdpdHMgUHR5IEx0ZDEbMBkGA1UEAwwSRGF0YSBTZXJ2
ZXIgVGVzdCAxMIIDRzCCAjoGByqGSM44BAEwggItAoIBAQCpe1QmJoHjagZFAxq9
vJcmpdfGVEN7zW/WroTn+RDc4jXro1aMHUmXi7rWrmMFDWxev7bP+xfZm5+Wib3Y
tHXYlZ0Td8fDBqLzAUQmE5ak6j5SkttYLLyO8BybnBC2R2WKRzzfKx3tzNhJC6Vf
dCoYRdNAh2OpLj37wzcHd6oZe3/yeuIpaR+uB8idT4wTGJJ0Grh5k9lhul3lpyfy
EVSKEb5IN9hfxWrWMLH9rKcau/5dbhAesA+ThK2kD9Mpl108TSfbw3Pec7uyxCmS
BeBdHtwxnuSPdjohmFe0Qj2DJz4YSHeFbg50KdxlPKnrlvJmscQk2Obmzh2viq0w
w4ixAiEAsKDTCz2TpKRU8dQ+ut8JZP7YAXbIIRuD/ehssoacv9ECggEBAIGp0LOT
tQjtSiIoKyPZVG148kFtyMOnOkg4U0ji6x+dZNLDFQLEDmkVAihLYyW5yM+a/CHb
58esH+hfXQ11saagveE4WMTODEoJyquidX2jenXMcLe8qmOAjuvC0InNVGUD1LRp
MD9V0NC8gG6C+2s0X3secE8NiBxb6Xl20N/DFfRR/qmfhDK4pOXyVw4dMoBbH+u8
Fz3BV4rZj44pD/vu2odRq6/c7mGp4VEfZPdALZ0mpwEyyvNFOKiR2vfp312xayte
iDV2Lam1LxItXhjp5LNJCPhtYaFvJIALxGnoL+b7bF1EwFHSp7bjxkJ3saEqLQYY
lr0aVktFv+72yxIDggEFAAKCAQBXJWtCbMVCq1O3u83fErBFL1Bdwn1wHWWDnSQL
QhtJ+8vww3xSMOvgEcsnbP+1ksLEYB4r7djZkkB13h9JIsjP1uHAv2dxJffcw3+E
mgKl2UiFdlEPlVqAv+4XvuNijTBC4lwkucZd+mgcdKE2+MJRnLtNsc5umZawNhhZ
BgXO/O9ANnj2V6tvpz5O9hXPC5WV+PrXXY3x222CdG0CpdmAFdUipLSEhFrGrLn0
cp4oOADbJeX3KuTx6P1+Lhjmd+PMXSuVvz4W1cxLCCWHAl+zCJItScF9L5/HOB5T
024LyZo2JRc8UOCkXQnxBYxU8YOGgU6OolnQAHv0a0eeG/IcMAsGCWCGSAFlAwQD
AgNIADBFAiABU71daIIQExWNy20flFo+aCtmdjaIx8FnAQdsje+0FQIhALI4ySv6
kA6jnvmerfi22Qr9oaU2SryvuED/cn6BJjAF
        </S100CE:certificate>
```

15-8.7 Creation of digital signatures by a Data Server

The Data Server creates a digital signature for the required data files using the DSA algorithm and their Private Key, see clause 15-8.4.

All files included in an S-100 Exchange Set must have their signatures encoded in either the S100_DatasetDiscoveryMetaData-digitalSignature or S100_SupportFileDiscoveryMetadata-digitalSignature elements.

The digitalSignatureReference field must be encoded "**DSA**".

The Data Server certificate must always be provided with a digital signature. It enables the OEM to authenticate the certificate using the SA public key and checking the certificate validity. The Data Server public key can be extracted from the certificate and used to authenticate the dataset file.

The individual id attributes can be used as a look-up by an OEM when a digital signature is defined. It reduces the need to repeat a Data Server certificate every time a signature is encoded.

The same XML elements for a Data Server certificate and digital signature defined in the Exchange Set catalogue are also used for digitally signing auxiliary files not included in the catalogue metadata; for example, catalogue and permit files. These are included in the S-100 XML Schemas and are self-contained with all necessary Data Server certificates included in them.

Since it is possible for Domain Coordinators (for example, IMO) to create Data Server certificates for participants of their domain, the following mechanism must be used to ensure the Data Client system can perform a certificate path validation:

1. The Data Server must always include the digital certificate of its Domain Coordinator to ensure the Data Client OEM has all the certificates required to perform a full certificate path validation without any external access.
2. When a Data Server certificate is defined in the catalogue metadata, it will include a data server ID and a reference to the issuer. The OEM should look up the issuer certificate and use it for Data Server authentication.
3. The OEM should verify the identity of the certificate issuer to verify the correct domain certificate to be used for Data Server certificate authentication. This is done prior to verification of the signature in accordance with DSA. All certificates in the Exchange Set shall be authenticated by the SA, either directly or through indirect authentication by one or more Domain Coordinators.

The digital signature is used in the catalogue metadata (and support file metadata) in two areas:

- The DSA-SHA256 digital signature of the data file, the R,S pair is embedded within the appropriate XML element according to the S-100 XML Schemas and base64 encoded; for example:

```
<digitalSignature id=”sig1” certificateRef="PRIMAR">
    MCwCFHNVdrZxQmKidddm70fgUTaEpr9SAhRYbnN23gM/hw/DlEffv8lpECL5QA==
</digitalSignature>
```

- The Data Server certificate (which remains constant). This is encoded as per clause 15-8.4 and should be embedded in the header of the catalogue metadata. This certificate provides the Public Key against which the digital signature (and the file content) is verified. The Data Server certificate is itself signed by the Scheme Administrator (or an intermediate Domain Coordinator) and it is the responsibility of the implementer to ensure that a separately installed root certificate from the SA is available on the implementing system. Data Server certificates should be authenticated prior to authentication of the dataset file.

The Data Server certificate only needs to be included in full a single time in the Exchange Set metadata. Since the certificate does not change it can be referred to by its "id" attribute when referenced by multiple digital signatures.

Another encoding of a digital signature is the PERMIT.SIGN file which holds a standalone signature of permit file content created by the Data Server issuing the permit. The PERMIT.SIGN file is a self-contained digital signature containing elements defining the file name, the digital signature and any certificates (and intermediate Domain Coordinator certificates) required.

Data Client systems shall verify the authenticity of the permit file to ensure the signature is valid and authenticated by the SA prior to installation of any permits.

15-8.8 Additional digital signatures

Additional digital signatures can be added by appending extra digital signature entries to the catalogue entry. This can express a list of certified identities signing an individual resource. This is an optional enhancement to the digitalSignature, the minimum being a single digitalSignature, verifying the content of a single resource (for example, a permit, catalogue, dataset or supplementary dataset file) against a named certificate.

- Additional digital signatures have their own XML type and can either sign the resource itself, or an existing signature of the resource.
- Additional digital signatures of the resource are appended to the exchange catalogue entry and have the same format as existing digital signatures. The dataStatus element denotes whether the signature refers to unencrypted, compressed or encrypted (and compressed) resources.
- Chains of digital signatures are implemented by use of a signatureRef attribute. A chained digitalSignature signs the content of another digital signature of the resource in the exchange catalogue. In this case the content signed is the ASN.1 byte array representing the R,S pair of the referenced signature.
- Each signature in the chain requires a valid certificateRef and an identifying “id” attribute.

These attributes are summarized in Table 15-10 below:

Table 15-10 – Additional digital signature attributes

Attribute	Purpose
id	Unique identifier of the digital signature value
certificateRef	The public key which the signature can be verified against. This is only optional if the signed public key is included in a digital signature element itself, otherwise it is mandatory
dataStatus	[For data signatures only] whether the signature is of an unencrypted resource, one which is compressed only (such as an archive of multiple resources) or encrypted (and compressed)

A full example, contained within a datasetDiscoveryMetadata element, follows. In this example the dataset discovery metadata specified a datafile. The first signature “s1” signs the dataset resource (no “ref” attribute is required), signature “s2” signs the encrypted data and signature “s3” signs signature s2.

```
[datasetDiscoveryMetadata entry]
  <S100XC:signature id="s1" certificateRef="PROD1" dataStatus="Unencrypted">
      MEUCIQCplrd+/Bb436FwXQWxgwxdcj9BhMN+EiMmZD4/6khpWwIgfLy70alp7pZSEeR27zThhQCl
OVA/ST01C+75Ond1Pu0=
  </S100XC:signature>
  <S100XC:additionalSignature>
    <S100_SE_SignatureOnData id="s2" certificateRef="RENC1" dataStatus="Encrypted">
MEYCIQCg+OdfUcfGJUxUKd53NmtGJ9jVOTACrKTRQM96KE0yCgIhAJ35u1aQjej7absi/V1lVOTS
DtKPUjxWAh+/DOWj+IOW
    </S100_SE_SignatureOnData>
  </S100XC:additionalSignature>
  <S100XC:additionalSignature>
    <S100_SE_SignatureOnSignature id="s3" certificateRef="DIST1" signatureRef="s2">
       MEUCIQCTGuSnqrbdQmO8ar4DdRGOjF8n5CI/9f/pGhDPeB2QhQIgMawaStd1wWJXiw1aDpz2JV/r
F9Hsx2txMN/3f2t8FIM=
    </S100_SE_SignatureOnSignature>
  </S100XC:additionalSignature>
```

15-8.9 Verifying Data Integrity and Digital Identity with an S-100 digital signature

Digital signature verification is an algorithm which operates on three independent pieces of data (all formatted in line with this Part of S-100):

1. Some **content** which requires validation (the format of this content is arbitrary);

2. A **Public Key**, suitably encoded. In the DSA algorithm adopted this Public Key is a single number together with a set of DSA parameters (three numbers);
3. A **signature**. In the DSA algorithm a signature is composed of two numbers; by convention these are referred to as R and S (an R,S pair).

A signature verification process identifies whether the R,S pair authenticate the content against the given Public Key. This can only result in a true or false result.

DSA digital signature verification achieves two results:

- **Authentication**: The implementing system verifies the Data Server Public Key ("**content**") and the signature in the Data Server certificate ("**signature**") against the SA Public Key (or Domain Coordinator) ("**Public Key**") to confirm that the supplier's Public Key in the certificate is valid and that the Data Server is a bona fide member of the S-100 Data Protection Scheme. If a Domain Coordinator is provided then the identity of the Domain Coordinator must also be checked against the SA Public Key.
- **Integrity Check**: The implementing system verifies the data signature ("**signature**") and the Data Server Public Key in the Data Server certificate ("**Public Key**") against the data file ("**content**"). This verifies the content of the data file.

If this validation check is successful then it proves that the data has not been corrupted in any way and that the identity of the Data Server within the dataset signatures is validated by the SAs identity as defined in the SA root certificate. The SA root certificate containing its public key must be installed separately on the end user system and is not packaged with the Exchange Set metadata.

15-8.10 MRN specifications

In order to support discoverability of Part XX Exchange Set resources the following MRN namespaces are defined by this Part of S-100. These are intended to be used to enable discovery of dataset supplementary resources by unique cryptographic hash or digital signature. The algorithm used to define the hash or signature is embedded in the MRN.

Tables 15-11 and 15-12 below show the specifications for digital signature and hash MRNs in S-100. All fields are mandatory and case-insensitive.

Table 15-11 – S-100 digital signature MRN

Name	Value	Example
Prefix	`urn:mrn:iho:s100:dsig`	
Algorithm	From digitalSignatureReference (Part XX 4a-5)	`dsa`
Value	Computed digital Signature value	**`MEQCIHVvkGrJl0joEqmS5PCmnJW4pydisZW5gpJGoU3CUeOVAiAZvuRA0y3QDLgnzJ8Il4oFX4U40BJ36UhRBVLUFfiVwQ==`**
Example	`urn:mrn:iho:s100:dsig:dsa:MEQCIHVvkGrJl0joEqmS5PCmnJW4pydisZW5gpJGoU3CUeOVAiAZvuRA0y3QDLgnzJ8Il4oFX4U40BJ36UhRBVLUFfiVwQ==`	

Table 15-12 – S-100 cryptographic hash MRN

Name	Value	Example
Prefix	`urn:mrn:iho:s100:hash`	
Algorithm	digitalSignatureReference (Part XX 4a-4.5)	`sha256`
Value	Computed cryptographic hash expressed as hexadecimal	**`a948904f2f0f479b8f8197694b30184b0d2ed1c1cd2a1ec0fb85d299a192a447`**

Example	urn:mrn:iho:s100:hash:sha256:a948904f2f0f479b8f8197694b30184b0d2ed1c1cd2a1ec0fb85d299a192a447

15-8.11 Exchange catalogue metadata and standalone schema element specification

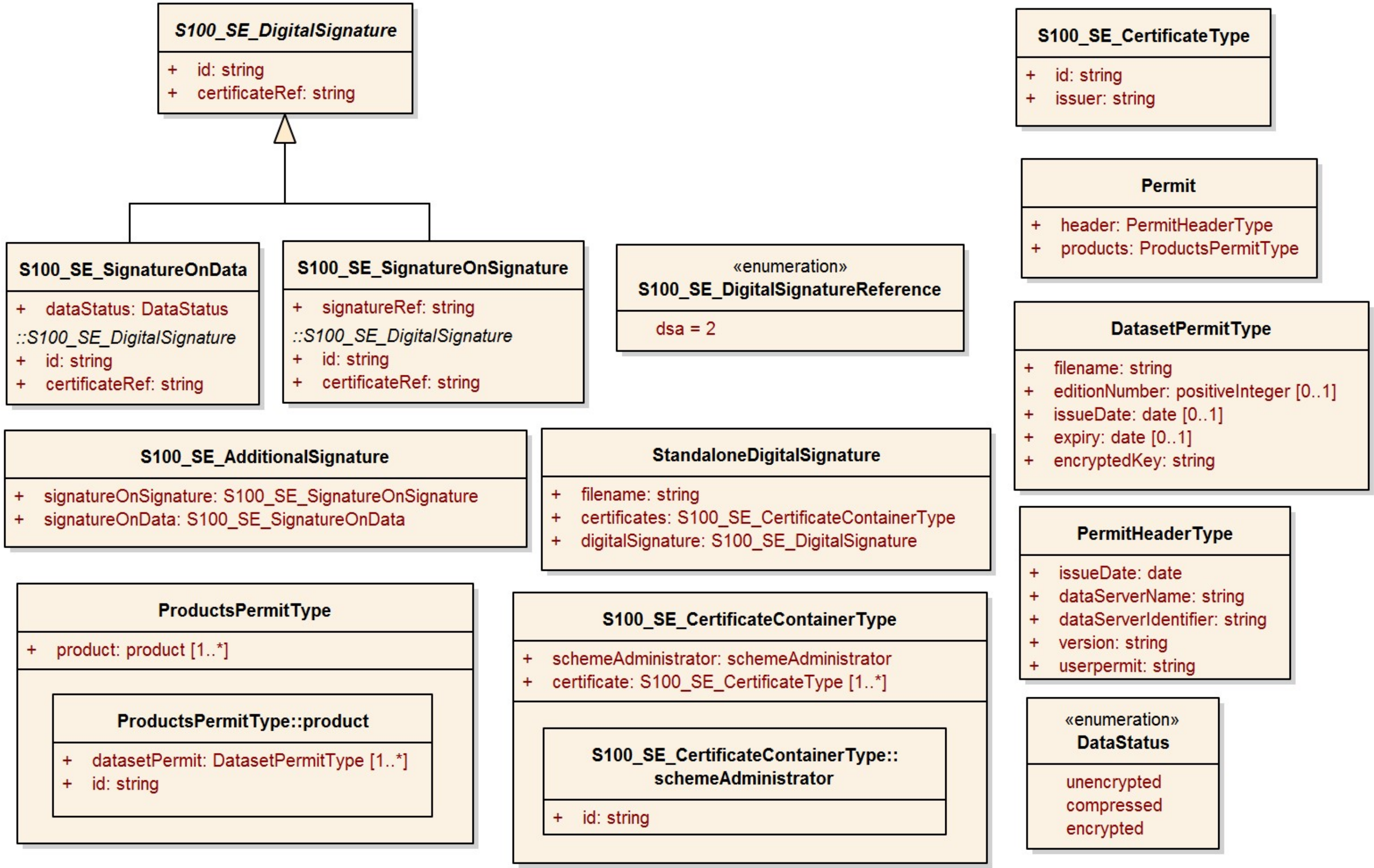


Figure 15-8 – Data protection – class details

15-8.11.1 S100_SE_CertificateContainerType

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_SE_CertificateContainerType	A set of signed public key certificates	-	-	Used in S-100 Part 17 Exchange Catalogues
Attribute	schemeAdministrator	The scheme administrator identity	0..1	CharacterString	The identity of the Scheme Administrator is contained in the “id” attribute of the schemeAdminstrator element. The scheme Adminstrator certificate is NOT included in catalogue metadata as it is independently verified by the implementing system
Attribute	certificate	A signed public key certificate	1..*	Base 64 encoded Character String	Conforms to X.509 encoding. Contains a digitally signed identifier of an entity

15-8.11.2 StandaloneDigitalSignature

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	StandaloneDigitalSignature	A single digital signature	-	-	-
Attribute	filename	The filename of the content signed	1	CharacterString	The filename of the resource signed
Attribute	certificates	Any certificates required to authenticate the signature against the SchemeAdministrator	1	S100_SE_CertificateContainerType	
Attribute	signature	A single digital signature	1	S100_SE_DigitalSignature	The signature of the file resource

15-8.11.3 S100_SE_DigitalSignature

The class S100_SE_DigitalSignature is realized as one of either S100_SE_SignatureOnData (a digital signature of a particular identified resource) or an additional digital signature defined using the class S100_SE_AdditionalSignature, each of which is either a S100_SE_SignatureOnData or S100_SE_SignatureOnSignature element as described in clause 15-8.8. S-100 Part 17 metadata thus allows for multiple digital signatures, a single mandatory S100_SE_SignatureOnData and any number of additional signatures, either of the data or other signatures.

15-8.11.4 S100_SE_SignatureOnData

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_SE_SignatureOnData		-	Base64 encoded digital signature value (clause 15-8.4)	-

Attribute	id	Identifier of the digital signature	1	CharacterString	Every signature entry has a unique identifier
Attribute	certificateRef	Signed Public Key	1	CharacterString	Identifier of the certificate against which the digital signature validates
Attribute	dataStatus	The digital signature	1	DataStatus	The digital signature value, calculated from the specified algorithm

15-8.11.5 S100_SE_SignatureOnSignature

Role Name	Name	Description	Mult.	Data Type	Remarks
Class	S100_SE_SignatureOnSignature		-	Base64 encoded digital signature value (Section 15-8.4)	-
Attribute	id	identifier of the digital signature	1	CharacterString	Every signature entry has a unique identifier
Attribute	certificateRef	Signed Public Key	1	CharacterString	Identifier of the certificate against which the digital signature validates
Attribute	signatureref	The digital signature referenced.	1		

15-8.11.6 DataStatus

Role Name	Name	Description	Remarks
Enumeration	DataStatus	The state of data when a digital signature is created	
Value	Unencrypted	The data is unencrypted and uncompressed	For example, supporting resources
Value	Encrypted	The data is compressed and encrypted	For example, copy protected datasets
Value	Compressed.	The data is compressed only	For example, archives of multiple resources

15-8.11.7 S100_SE_DigitalSignatureReference

Role Name	Name	Description	Code	Remarks
Enumeration	S100_SE_ DigitalSignatureReference	A reference to a cryptographic algorithm used in an implementation of Part 15		Only DSA is currently used in implementations of S-100 for file based transfer of data to ECDIS. Other values are included for interoperability with other implementations by external standards. See clause 15-8.4
Value	RSA		1	RSA with key length >= 2048 bits

Value	**DSA**		**2**	**DSA with key length >= 2048 bits**
Value	ECDSA		3	ECDSA with key length >= 224 bits.
Value	ECDSA-224-SHA2-224		4	224 bits ECDSA with SHA2-224 hashing
Value	ECDSA-224-SHA3-224		5	224 bits ECDSA with SHA3-224 hashing
Value	ECDSA-256-SHA2-256		6	256 bits ECDSA: SHA2-256
Value	ECDSA-256-SHA3-256		7	256 bits ECDSA: SHA3-256
Value	ECDSA-384-SHA2		8	384 bits ECDSA: SHA2-384
Value	ECDSA-384-SHA3		9	384 bits ECDSA: SHA3-384
Value	AES-128		10	AES 128 bit keys
Value	AES-192		11	AES 192 bit keys
Value	AES-256		12	AES 256 bit keys

15-9 Glossary of S-100 Data Protection Scheme and computing terms

For a list of general abbreviations used throughout S-100, see Part 0, clause 0-2. For a list of general terms and definitions used throughout S-100, see Annex A.

Table 15-13 – S-100 Data Protection Scheme terms

AES	Advanced Encryption Standard, encryption algorithm used in the scheme
Data Permit	File containing encrypted product keys required to decrypt the licensed products. It is created specifically for a particular user
Data Client	Term used to represent an information consumer receiving the encrypted information. The Data Client will be using a software application (for example ECDIS) to perform many of the operations detailed within the scheme.
Data Server	Term used to represent an organization producing encrypted data files or issuing Dataset Permits to end-users
M_ID	The unique identifier assigned by the SA to each manufacture. Data Servers use this to identify which M_KEY to use when decrypting the Userpermit
M_KEY	System manufacturer’s unique identification key provided by the Scheme Administrator to the OEM. It is used by OEMs to encrypt the HW_ID when creating a userpermit
HW_ID	The unique identifier assigned by an OEM to each implementation of their system. This value is encrypted using the OEM’s unique M_KEY and supplied to the data client as a userpermit. This method allows data clients to purchase licences to decrypt datasets
PKCS	Public Key Cryptography Standards
IV	Initialization Vector used by the AES-CBC encryption algorithm
SA	Scheme Administrator. IHO is responsible for maintaining and coordinating all operational aspects and documentation of the Protection Scheme
SHA	Secure Hash Algorithm
SSK	Self Signed Key (Self Signed Certificate File)
User Permit	Encrypted form of HW-ID uniquely identifying the Data Client system

Table 15-14 – Computing terms

CRC	Cyclic Redundancy Check
XOR	Exclusive OR

Page intentionally left blank

S-100 - Part 16

Interoperability Catalogue Model

Page intentionally left blank

Contents

Page intentionally left blank

16-1 Scope

Users of S-100-based data products are likely to receive and use different S-100-based data products simultaneously and those data products are likely to contain features which may have some common elements or be different expressions of similar underlying concepts. Other data layers such as radar overlays may also be present. The smooth interoperation and harmonized, user-friendly graphical presentations of these various products is necessary. This Part defines a framework for creating rules for the interoperation of S-100 data products, including harmonized graphical presentations and handling of alarms and indications.

This framework can be used to establish system specific rules which are contained in an Interoperability Catalogue, a type of meta-product that describes how groups of products are to be used and displayed simultaneously.

16-2 Conformance

This Part is not a profile of an ISO or other standard because the concept of S-100 interoperability as used in this Part is unique to S-100. The S-100 Interoperability Model is a specialization of the CT_Catalogue class defined in ISO 19139, as implemented in ISO 19115-3.

16-3 Normative References

RFC 2141 URN Syntax. Internet Engineering Task Force (IETF), May 1997.

IMO MSC.1/Circ.1512 Guideline on Software Quality Assurance and Human-Centred Design for e-Navigation.

ISO 19115-1 ISO 19115-1:2014, Geographic information – Metadata – Part 1 - Fundamentals. As amended by Amendment 1, 2018.

ISO 19115-3 ISO/TS 19115-3:2016, Geographic information - Metadata - XML Schema implementation for fundamental concepts.

XPath XML Path Language (XPath) 3.1 - W3C Recommendation 21 March 2017 (World-Wide Web Consortium – W3C). URL: https://www.w3.org/TR/2017/REC-xpath-31-20170321/ (Retrieved 2017-12-08).

16-4 Context

This Part provides a framework for specifying machine-readable rules governing the interaction of data products in an S-100 compatible systems. A system that utilises different data products needs a prescribed algorithm to determine how these different data products should interact within the system. Interaction in this context means the manner in which the simultaneous display of data products affects the appearance of the combined display compared to the appearance of each product displayed in isolation. Interoperability of products includes issues such as ensuring that additional information overlay products do not obscure other significant information, or managing the presence of same or similar feature instances in different products, which can leave the user questioning which instance is the most accurate and appropriate for use.

The rules governing interoperability are not expected to be the same for all product interactions and therefore must be managed for groupings of products that are expected to be used together under specific circumstances, for example when performing specific navigational tasks. Rules that manage the information according to these principles, are captured in a machine-readable XML format, so that systems can be updated with new sets of rules in events such as additional products becoming available or alterations to the existing combinations becoming necessary. The set of rules is called an Interoperability Catalogue.

16-4.1 Overview of approach to implementing interoperability

Interoperability processing works in combination with regular portrayal processing (see S-100 Part 9 - Portrayal). Depending on the system architecture designed by developers of system software, it may for example function as a pre-processing or post-processing stage to regular portrayal processing.

An Interoperability Catalogue basically describes a transformation from an input stream of feature data to an output stream of prioritized feature data. The input stream consists of feature data from S-100 based datasets (either in the form of feature objects from the input data, or feature objects transformed to drawing instructions by portrayal processing). The output feature stream consists of feature data that may contain some input features in their original form; combine versions of other features; and remove other features from the stream altogether. Features in the output stream also have assigned (or revised) priorities in relation to other features, depending on feature type.

16-4.2 Overview of an Interoperability Catalogue

An Interoperability Catalogue is a collection of rulesets for filtering and/or combining feature objects from different input streams corresponding to different data products into feature objects in a combined output stream. The Interoperability Catalogue provides means of describing the conditions under which a ruleset is active – that is, each ruleset applies to a particular combination of products.

Within each ruleset, there may be rules whose antecedents specify the combinations of features from different products to which they apply, and what the result of applying the rule is; that is, given a particular combination of features in the input stream, what feature or features should be emitted into the output stream.

The interoperability processor applies the rules which are assigned to the loaded combination of data products, and outputs a stream of feature data which contains both original feature instances and any new instances, when applicable, which conform to new hybrid feature types that are defined in a Hybrid Feature Catalogue. The selection process is shown in Figure 16-1, and consists only of selecting the applicable predefined combination (PDC) that corresponds to the user or system settings and which lists exactly the currently loaded products. There should be only one applicable PDC after these steps; however, if there is a tie it must be broken by external tie-breaking methods such as allowing the user to select a combination based on the use conditions and description attributes of the PDC.

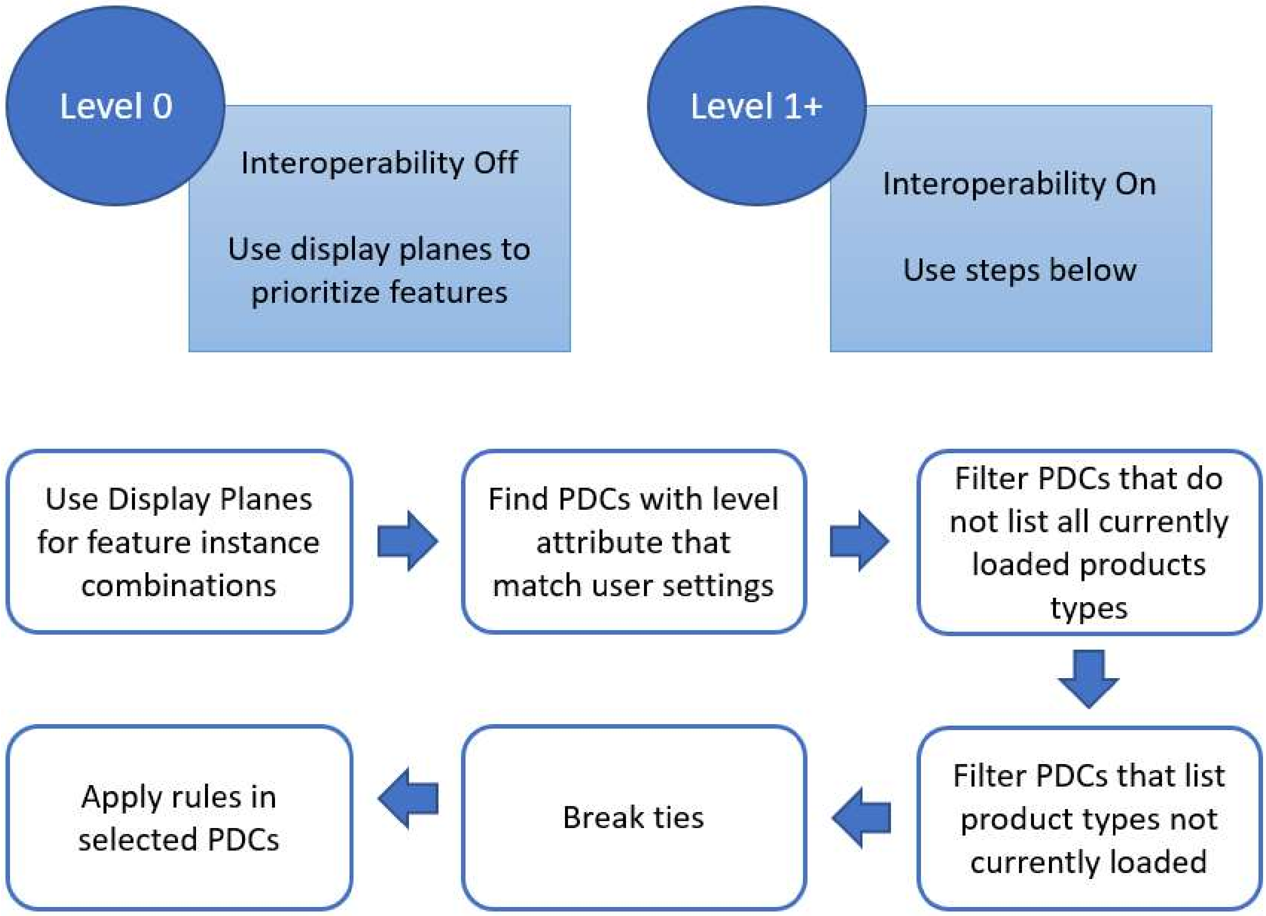


Figure 16-1 - Process for selecting rules

Product combinations are lists of data products. When associated with a ruleset or an individual rule, it means that the ruleset or single rule applies to the combinations of the data products listed. Each product combination may be assigned an interoperability level, if implemented, and descriptive attributes which indicate its purpose and applicability to the human end-user and catalogue developer. Levels are used to break up the functionality of the Interoperability Catalogue model into logical modules as appropriate for the use case. If the use case does not call for breaking up the functionality of the Interoperability Catalogue model only a single level 1 is required.

Display planes act as a means of layering features in the end-user's display. Each feature in the output stream is assigned to a display plane. Each plane is assigned a display order relative to other planes. The Interoperability Catalogue uses display planes for interleaving features from different products, with features in higher priority planes overlying those in lower priority planes wherever they overlap. In the case of coincident or overlapping symbols, the implementer may use appropriate methods to avoid displaying partial symbols, or "grafting" part of lower-layer symbols onto symbols in upper layers.

The Interoperability Catalogue allows suppression of feature classes where the presence in one product of features of a given feature type causes the removal of all features of a specified feature type from another specified data product. This is supposed to be used when the two feature types represent the same kind of data but one data product is preferred over the other. For example, level of detail in the preferred data product, additional feature characteristics in the preferred product, etc.

The Interoperability Catalogue also allows suppression of feature instances whereby only feature instances meeting specified conditions are suppressed in favour of feature instances from a different product. The conditions are described in terms of specific characteristics such as attribute values or combinations of values of different attributes, or the use of specific classes of spatial primitives such as all point features of the specified type.

EXAMPLE: Restricted area features from type A datasets with category attribute = (nature reserve), (bird sanctuary), (seal sanctuary), (ecological reserve), or (coral sanctuary) are suppressed in favour of restricted area features from type B datasets due to more details.

Advanced interoperability functionality includes hybridization of features. Hybridization consists of combining feature data from different products in the input stream into a new type in the output stream – new in the sense that the output feature type is not defined in any Feature Catalogue of the input products. Such combined types are intended for producing resultant features with enhanced characteristics, for example by enhancing the attribute set from one product with additional attributes derived from another data product.

EXAMPLE: water level information from one product is combined with bathymetry in another and with high definition bathymetry in a third product to create go and no-go areas that increase and decrease with changing water level.

The hybrid feature is defined in a hybrid Feature Catalogue (must comply with Part 5) and its portrayal is defined in a hybrid Portrayal Catalogue (must comply with Part 9) that is bundled with the Interoperability Catalogue in an exchange set.

16-4.3 Overview of processing

Interoperability processing can either precede or follow portrayal processing (except rendering, which converts feature data into graphics and is necessarily the step just before actual display). A mixed processing model, where interoperability processing is done both before and after portrayal processing, is also possible.

- Interoperability before regular portrayal processing: Feature data from S-100-based datasets is an input to the interoperability processor, along with the Interoperability Catalogue and context parameters. The interoperability processor filters and interleaves feature data according to the Interoperability Catalogue and interoperability level selected by the user and passes the resultant feature data to the portrayal processor, which uses the Portrayal Catalogue for individual products to generate drawing instructions for the display processor.
- Interoperability after regular portrayal processing: Feature data from S-100-based datasets flows to the portrayal processor. The portrayal processor transforms them into drawing instructions. The drawing instructions flow to the interoperability processor. The interoperability processor filters and interleaves the drawing instructions according to the Interoperability Catalogue and interoperability level selected by the user and passes the resultant drawing instructions to the display processor.

The two processing options described above are shown in Figure 16-2.

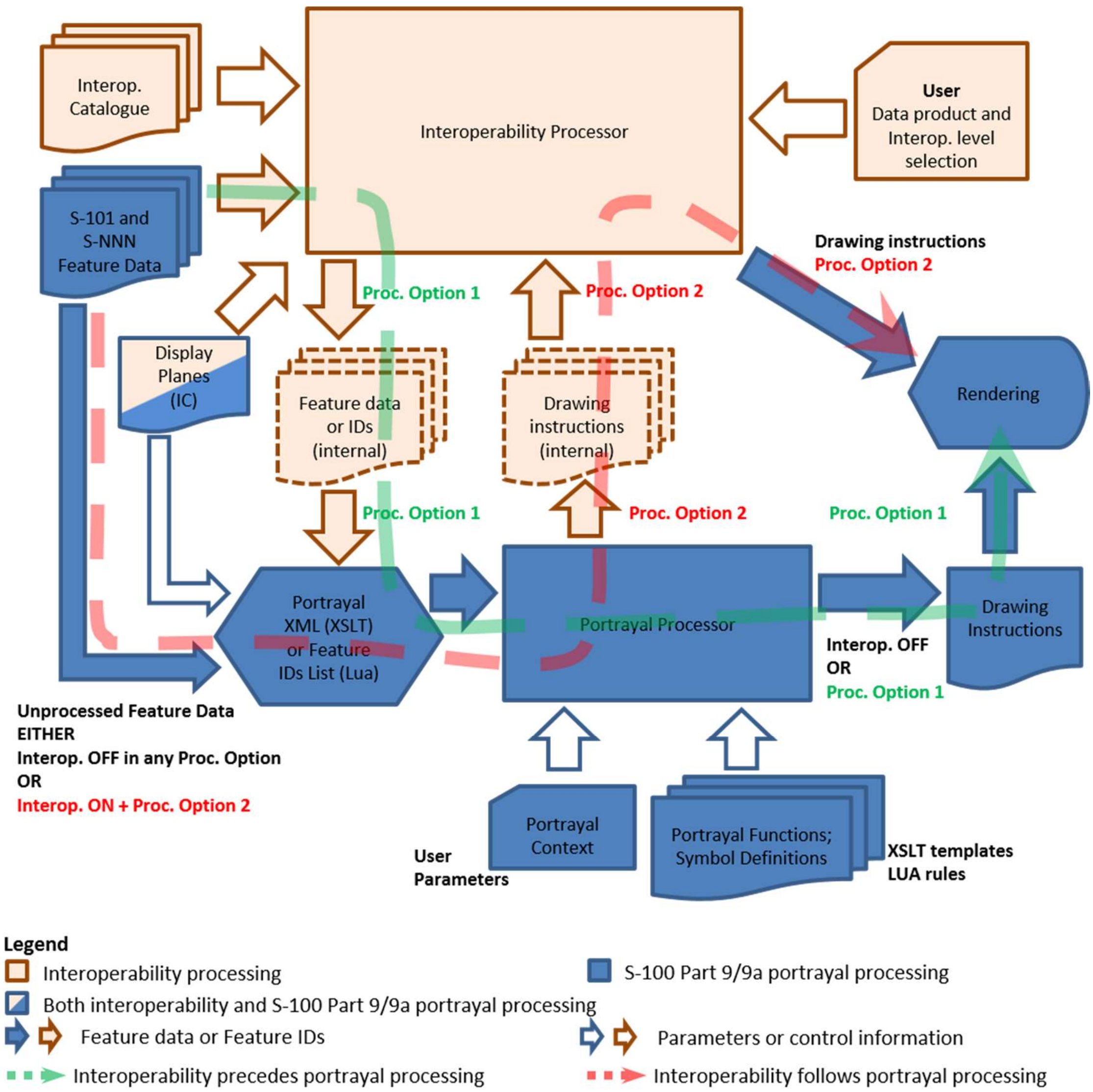


Figure 16-2 - Basic interoperability processing

In any level of processing except Level 0 (interoperability off), data products to be loaded are selected by the system according to the list in the predefined combination selected by the user selection from among those listed in the catalogue.

Feature data from products not listed in the Interoperability Catalogue are passed through to portrayal processing as described in S-100 Part 9 (stage *Portrayal Processing*) without any intermediate stages in interoperability processing, and displayed by ordinary S-100 portrayal processing according to their individual Portrayal Catalogues.

16-4.4 Interoperability Catalogue Data Model

16-4.4.1 Introduction

The Interoperability Catalogue specifies the relative display prioritization of feature types and feature instances, as defined in individual Product Specifications, in relation to other feature types and feature instances, which may be defined in any of the data products declared to be within the scope of the Interoperability Catalogue.

An Interoperability Catalogue describes display planes, predefined combinations, feature instance and feature layer suppression rules.

The Interoperability Catalogue (IC) utilizes the ISO **CT_Catalogue** class defined in ISO 19139 (implemented in ISO 19115-3) as a super-type for header information. The body of the Interoperability

Catalogue consists of subsections encoding the rules for display planes, feature priorities, feature interleaving, and available predefined combinations:

- display planes, indicating order of planes, viewing group, and drawing priority;
- predefined combinations and operations on feature types or feature instances for each combination;

An Interoperability Catalogue must be an XML document which conforms to the Interoperability Catalogue Schema which can be downloaded from the IHO website. Figure 16-3 shows the Interoperability Catalogue model.

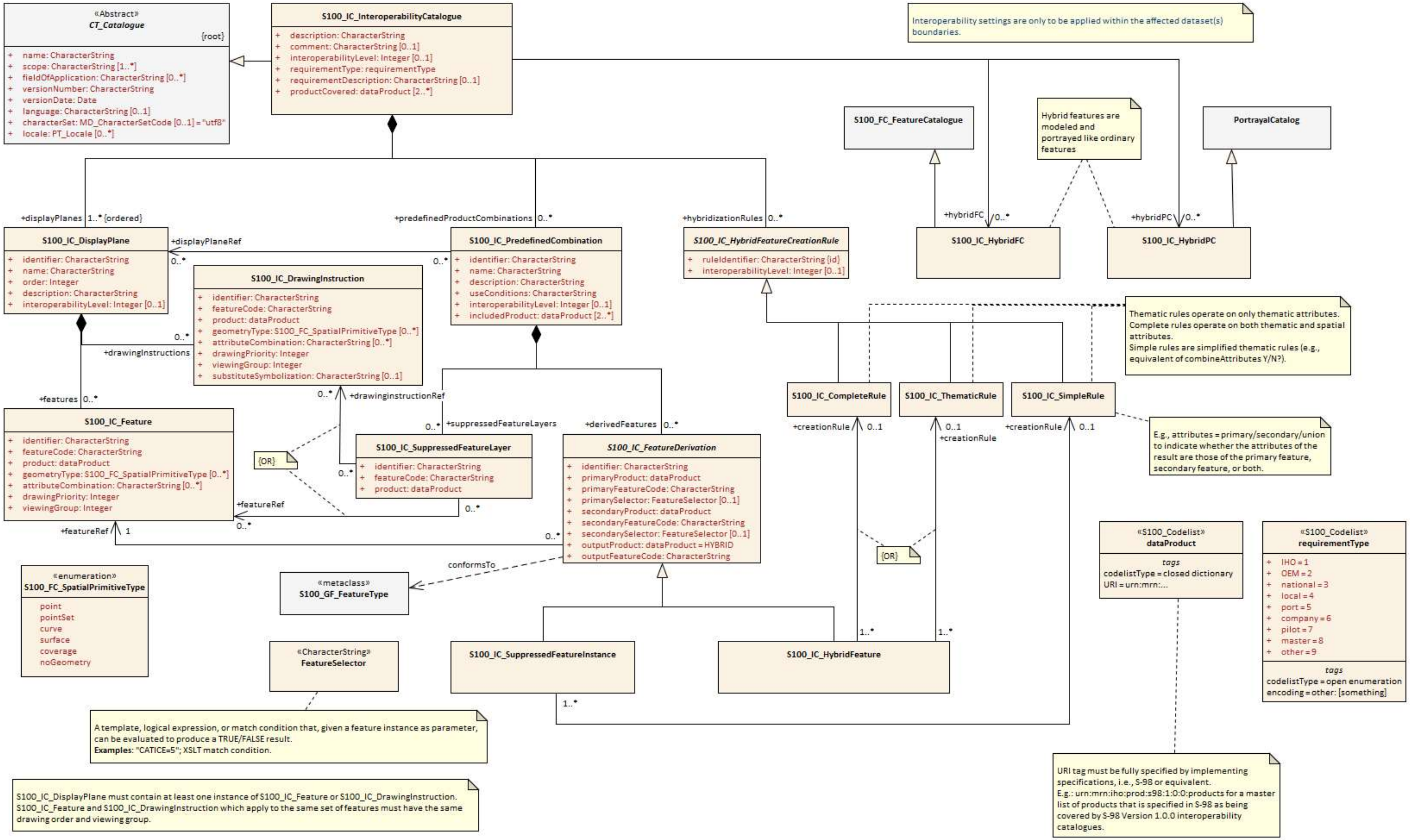


Figure 16-3 - Interoperability Catalogue Model

16-4.4.2 Interoperation conceptual types

The following clauses describe the different conceptual elements that may be used in an Interoperability Catalogue.

16-4.4.2.1 S100_IC_InteroperabilityCatalogue

An Interoperability Catalogue contains operations and rules for the interoperation of a set of S-100-based data products.

Table 16-1 S100_IC_InteroperabilityCatalogue

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_InteroperabilityCatalogue	Class that holds header information about an Interoperability Catalogue	--	--	Specialization of CT_Catalogue (ISO 19115-3)
Attribute	description	Description of the catalogue	1	CharacterString	
Attribute	comment	Any additional comments	0..1	CharacterString	
Attribute	interoperabilityLevel	The highest level of interoperability functionality encoded within an instance of this type	0..1	Integer	
Attribute	requirementType	The type of authority or requestor responsible for the specifications, rules, or requirements based on which this catalogue was prepared	1	Enumeration	
Attribute	requirementDescription	Description of the source of the requirements or specifications upon which this catalogue is based. This might be the name of the country, company, OEM, port, pilot, etc	1	CharacterString	
Attribute	productCovered	The products covered by this catalogue	2..*	dataProduct	
Attribute	name	The name for the catalogue	1	CharacterString	Inherited from CT_Catalogue
Attribute	scope	Subject domain of the catalogue	1..*	CharacterString	Inherited from CT_Catalogue
Attribute	fieldOfApplication	Description of the use to which this catalogue may be put	0..*	CharacterString	Inherited from CT_Catalogue
Attribute	versionNumber	The version number of the Product Specification	1	CharacterString	Inherited from CT_Catalogue

Attribute	versionDate	The version date of the Product Specification	1	Date	Inherited from CT_Catalogue
Attribute	language	The language used for this catalogue	0..1	CharacterString	Inherited from CT_Catalogue
Attribute	locale	provides information about alternatively used localized character strings	0..1	PT_Locale (ISO 19115-1)	Inherited from CT_Catalogue
Attribute	characterSet	Character set used in the catalogue	0..1	MD_CharacterSetCode (ISO 19115-1)	Inherited from CT_Catalogue must have value=utf8
Composition	displayPlanes	Container for one or more S100_IC_DisplayPlane elements	1..*	<sequence>S100_IC_DisplayPlane	Ordered list of one or more S100_IC_DisplayPlane elements
Composition	predefinedProductCombinations	Container for predefined product combinations and the interoperability operations for each	0..*	<sequence>S100_IC_PredefinedCombination	Sequence of S100_IC_PredefinedCombination elements
Composition	hybridizationRules	Container for hybridization rules	0..*	<sequence>S100_IC_HybridizationRule	
Role	hybridFC	Reference to hybrid Feature Catalogue used by operations in this Interoperability Catalogue	0..*	CharacterString	
Role	hybridPC	Reference to hybrid Portrayal Catalogue used by operations in this Interoperability Catalogue	0..*	CharacterString	

16-4.4.2.2 S100_IC_DisplayPlane

A display plane element acts as a container for display information for specified feature classes. The display order for the plane as a whole is provided in the S100_IC_DisplayPlane element. All the types within an instance of S100_IC_DisplayPlane have the same display order (encoded in attribute order) relative to feature types in another instance of S100_IC_DisplayPlane. Instances of display plane can be characterized by interoperability level, which allows the encoding of different sets of operations depending on how tightly integrated the user desires the products to be on the resultant display.

Assigning feature types to display planes enables the interleaving of feature layers during portrayal by indicating the display plane, priority, and rendering order of the types assigned to a display plane. **S100_IC_DisplayPlane** assigns subsets of feature types to display planes and defines the viewing group, drawing priority, and significance for each feature type in the plane. An **S100_IC_DisplayPlane** element may include more than one feature type.

A feature type may be referenced in more than one **S100_IC_DisplayPlane**, but the entries in different display planes must be distinguished by different attribute-value combinations or spatial primitives so that the actual instances of features are partitioned unambiguously between different display planes.

The portrayal of feature types not mentioned in any S100_IC_DisplayPlane component is undefined until ordinary portrayal processing.

An instance of **S100_IC_DisplayPlane** must contain at least one instance of **S100_IC_Feature** or **S100_IC_DrawingInstruction**. An instance of **S100_IC_DisplayPlane** may contain both **S100_IC_Feature** and **S100_IC_DrawingInstruction**, subject to the constraint below.

For **S100_IC_Feature** and **S100_IC_DrawingInstruction** with the same combination of [featureCode, product, geometryType, and attributeCombination] and in the same **S100_IC_DisplayPlane** container:

- **S100_IC_Feature.drawingPriority** and **S100_IC_DrawingInstruction.drawingPriority** must have the same values.
- **S100_IC_Feature.viewingGroup** and **S100_IC_DrawingInstruction.viewingGroup** must have the same values.

Table 16-2 S100_IC_DisplayPlane

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_DisplayPlane	Each display plane identifies all features and their drawing priority within the plane. The order in which display planes are rendered is also given.	--	--	Composition component of S100_IC_InteroperabilityCatalogue, container displayPlanes
Attribute	identifier	Unique identifier of the display plane	1	CharacterString	Must be unique
Attribute	name	Name of display plane	1	CharacterString	Under radar, over radar, etc.
Attribute	order	Used to sort the rendering order of display planes. Display planes with larger values are drawn above those with lower values.	1	Integer	Refer S-100 Part 2, clause 2b-4.2.23 and Part 9, clauses 9-11.1.5, 9-13.3 & 9-13.3.20 Positive: Above RADAR Zero: Reserved for RADAR Negative: Below RADAR
Attribute	description	description of the display plane	1	CharacterString	
Attribute	interoperabilityLevel	The highest level of interoperability functionality encoded within an instance of this type	0..1	Integer	
Composition	features	Container for S100_IC_Feature elements	0..*	<sequence>S100_IC_Feature	At least one S100_IC_Feature or S100_IC_DrawingInstruction element must be included in a display plane element
Composition	drawingInstructions	Container for S100_IC_DrawingInstruction elements	0..*	<sequence>S100_IC_DrawingInstruction	At least one S100_IC_Feature or S100_IC_DrawingInstruction element must be included in a display plane element

16-4.4.2.3 S100_IC_Feature

The **S100_IC_Feature** element describes the display parameters for all features of a specific feature type in a specific product. The **S100_IC_Feature** element determines the order of drawing the feature type identified by its featureCode attribute relative to other feature types in the same display plane. It also specifies the viewing group to which the feature is assigned. Its applicability can be optionally restricted to a subset of instances of the feature type by additional attributes that specify the type of spatial primitive and indicate specific values of thematic attributes.

The **S100_IC_Feature** element in Interoperability Catalogues is similar in operation to the layering and priority aspects of the **DrawingInstruction** element in Portrayal Catalogues (see S-100 Part 9 - Portrayal), and therefore has attributes that are equivalent to some of the attributes and roles of the Portrayal Catalogue element. Where there is an exact correspondence with a Portrayal Catalogue element, the element in the Interoperability Catalogue element supersedes the Portrayal Catalogue element. The correspondences are summarized in Table 16-19 at clause 16-5.

Table 16-3 S100_IC_Feature

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_Feature	Information that guides the relative layering and drawing priority of feature types during portrayal	--	--	
Attribute	identifier	Internal identifier of the catalogue element	1	CharacterString	
Attribute	featureCode	The code assigned to the feature type in Feature Catalogue for the product indicated in the product attribute	1	CharacterString	
Attribute	product	A data product	1	dataProduct	
Attribute	geometryType	The type of spatial primitive that indicates the location	0..*	S100_FC_SpatialPrimitiveType	
Attribute	attributeCombination	Describes attribute-value filters to be applied to the specified features	0..*	CharacterString	See Clause 16-4.2.3
Attribute	drawingPriority	Drawing priority of feature type in the display plane	1	Integer	Refer S-100 Part 9, clause 9-11.2.2
Attribute	viewingGroup	The viewing group of the feature type	1	Integer	Refer S-100 Part 9, clause 9-13.3

NOTE: **S100_IC_Feature** and **S100_IC_DrawingInstruction** elements operate in essentially the same way as far as assignment of drawing priority, and display planes is concerned. They differ in that **S100_IC_DrawingInstruction** provides an optional attribute to substitute the symbolization elements of the drawing instruction. **S100_IC_Feature** should be used for Interoperability Catalogues that are designed for systems where interoperability processing precedes the generation of drawing instructions. **S100_IC_DrawingInstruction** should be used for Interoperability Catalogues that are designed for systems where interoperability processing precedes the generation of drawing instructions. It should also be used in all catalogues where substitution of symbolization is necessary.

16-4.4.2.4 S100_IC_DrawingInstruction

Drawing instructions specify the display order used by the rendering engine in producing the portrayal output of a given feature type/geometric primitive type/attribute value combination. The **S100_IC_DrawingInstruction** element determines the order of drawing the feature type identified by its **featureCode** attribute relative to other feature types in the same display plane. The applicability of an **S100_IC_DrawingInstruction** to feature types can be further restricted by the type of spatial primitive and values of thematic attribute, using **geometryType** and **attributeCombination** attributes of the **S100_IC_DrawingInstruction** class.

The **S100_IC_DrawingInstruction** element in Interoperability Catalogues is similar in operation to the layering and priority aspects of the **DrawingInstruction** element in Portrayal Catalogues (see S-100 Part 9 - Portrayal), and therefore has attributes that are equivalent to some of the attributes and roles of the Portrayal Catalogue element. Where there is an exact correspondence with a Portrayal Catalogue element, the element in the Interoperability Catalogue element supersedes the Portrayal Catalogue element. The correspondences are summarized in Table 16-19 in clause 16-5. (Definitions of the Portrayal Catalogue attributes are provided in S-100 Part 9).

The **S100_IC_DrawingInstruction** element contains the **substituteSymbolization** attribute that allows substitution of symbolization instructions generated by portrayal processing. Note that the display instruction XML elements defined in the presentation XML Schema S-100 Part 9 (**pointInstruction**, **lineInstruction**, etc.) cannot be used directly because, being extensions of the base type **DrawingInstruction** in that Schema, they: (a) reference individual feature and spatial instances, and (b) contain viewing group, display plane, and drawing priority as mandatory elements, which would be redundant.

Table 16-4 S100_IC_DrawingInstruction

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_DrawingInstruction	Information that guides the relative layering and drawing priority of drawing instruction during portrayal.	--	--	
Attribute	identifier	Internal identifier of the instruction group	1	CharacterString	
Attribute	featureCode	The code assigned to the feature type in Feature Catalogue for the product indicated in the product attribute	1	CharacterString	Corresponds to the feature reference for drawing instructions in S-100 Part 9
Attribute	product	A data product	1	dataProduct	
Attribute	geometryType	The type of spatial primitive that indicates the location	0..*	S100_FC_SpatialPrimitiveType	
Attribute	attributeCombination	Describes attribute-value filters to be applied to the specified features	0..*	CharacterString	See Clause 16-4.4.3
Attribute	drawingPriority	The drawing priority of the group	1	Integer	Refer S-100 Part 9, clause 9-11.2.2
Attribute	viewingGroup	The viewing group of the feature type	1	Integer	Refer S-100 Part 9, clause 9-13.3

Attribute	substituteSymbolization	Substitute for the symbolization content of drawing instructions. This can be any element of the drawing instruction not defined in the abstract class DrawingInstruction defined in S-100 Part 9, clause 9-11.2, but defined in the relevant descendant of that class	0..1	CharacterString	The string must consist of one or more XML fragments constructed according to the Presentation Schema in S-100 Part 9 or the equivalent in a non-XML syntax. A CDATA section may be used to avoid the explicit encoding of character entities for special characters

NOTE: Even if the Presentation Schema in S-100 Part 9 is used, specific code may need to be provided to validate the content of the **substituteSymbolization** attribute instead of depending on purely XML Schema validation. The content of this attribute is not prescribed by this specification and may be a fragment of XML, or interpretable code or rules, etc., in a non-XML syntax. It may be enclosed in a `<![CDATA[ … ]]>` section so that XML validators treat it as character data instead of XML. If the content is XML, it must be well-formed (for example have balanced opening and closing tags).

16-4.4.2.5 S100_IC_SuppressedFeatureLayer

Each instance of this element identifies a feature type in a specific data product.

Table 16-5 S100_IC_SuppressedFeatureLayer

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_SuppressedFeatureLayer	Describes operations for suppressing all instances of a feature type in one product by features from another product	--	--	
Attribute	identifier	Internal identifier of the catalogue element	1	CharacterString	
Attribute	featureCode	Feature type code in the FC for the product mentioned in attribute **product**	1	CharacterString	
Attribute	product	The data product for the type being replaced	1	dataProduct	
Role	featureRef	References to replacement features' display specifications in the display planes section of the Interoperability Catalogue	0..*	<reference>S100_IC_Feature	Replacement by multiple feature types is intended for associated feature types, for example different feature types that make up a traffic separation scheme.
Role	drawingInstructionRef	Reference to S100_DrawingInstruction element	0..*	<reference>S100_IC_DrawingInstruction	

NOTE: Only one of featureRef or drawingInstructionRef can be used in an instance.
NOTE: If both feature and drawing instruction references are empty, the type is suppressed without being replaced.

16-4.4.2.6 S100_IC_PredefinedCombination

A predefined combination element defines a collection of data products for which a common set of interoperability operations have been defined in the Interoperability Catalogue. Instances of predefined combinations can be characterized by interoperability level, which allows the segmentation of different sets of interoperability operations depending on how tightly integrated the user desires the products to be on the resultant display, see Clause 16-8 for more details.

Predefined combination element can specify the following types of interaction between its listed products.

- Operations on selected instances of a feature type or conversion of input feature data into new feature data involving only thematic attributes.
- Operations involving operations on spatial attributes and possibly thematic attributes as well.

The simplest operations on instances are replacement of selected instances from one product by selected instances from another product. These are described by associated **S100_IC_SuppressedFeatureInstance** elements. More complex operations, including conversion of input feature instances into new features (hybridization) are described by associated **S100_IC_HybridFeature** elements. The replacement and hybridization rules are described in Clause 16-6.

Predefined combinations can be linked to **S100_IC_DisplayPlane** elements by means of references in the **S100_IC_PredefinedCombination** elements.

Table 16-6 S100_IC_PredefinedCombination

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_PredefinedCombination	Pre-defined combinations are identifiable pre-set collections of recommended and optional S-NNN data products which are expected to be loaded by the user under specific conditions or for specified tasks. Each pre-defined combination is basically a package of data products, display priorities, context parameters, user settings, Portrayal Catalogues, etc.	--	--	Composition component of S100_IC_InteroperabilityCatalogue
Attribute	identifier	Identifier of the predefined combination	1	CharacterString	For example, a sequence number, UUID or URN unique to the PDC in the catalogue. May be globally unique, but must be unique within the catalogue at least
Attribute	name	Name of combination	1	CharacterString	
Attribute	description	Brief description of combination	1	CharacterString	
Attribute	useConditions	Conditions for which the combination is designed	1	CharacterString	
Attribute	interoperabilityLevel	The highest level of interoperability functionality encoded within an instance of this type	0..1	Integer	

Attribute	includedProduct	Products loaded in this combination and referenced by operations and rules that apply to this combination	2..*	dataProduct	A combination must use at least 2 data products
Role	displayPlaneRef	Reference to an S100_IC_DisplayPlane element in this Interoperability Catalogue	0..*	<reference>S100_IC_DisplayPlane	
Composition	derivedFeatures	Container for S100_IC_SuppressedFeatureInstance or S100_IC_HybridFeature elements (concrete specializations of S100_IC_FeatureDerivation)	0..*	<sequence> of sub-classes of S100_IC_FeatureDerivation	
Composition	suppressedFeatureLayers	Container for S100_IC_SuppressedFeatureLayer	0..*	<sequence> S100_IC_SuppressedFeatureLayer	

NOTE: A system can allow the user to initiate the loading of multiple data products and activate multiple parameter settings as a single action, by selecting from a list of pre-defined combinations, instead of loading and unloading individual data products.

16-4.4.2.7 S100_IC_FeatureDerivation

S100_IC_FeatureDerivation is an abstract super-class for different types of feature hybridization operations. Individual primary and secondary inputs are suppressed from being rendered and only the resulting derived feature is added to the data stack.

The resulting derived feature does not need to have any hybrid characteristics, that is, one restricted area replaced with another restricted area will use the regular PC/FC of the primary product. However, if the result feature needs to be supported by any custom FC or PC elements, they must be defined under hybrid FC and hybrid PC accordingly.

A rule for creating the feature must be described in the rules section

Table 16-7 S100_IC_FeatureDerivation

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_FeatureDerivation	Derived features are created by consolidating features from 2 or more different products into one final view, so the changes can include geometry, attribution and/or portrayal (depending on the interoperability level)	--	--	Abstract class.
Attribute	identifier	Internal identifier of the catalogue element	1	CharacterString	
Attribute	primaryProduct	One of the two interoperating data products	1	dataProduct	
Attribute	primaryFeatureCode	Feature type code in the FC for the product mentioned in primaryProduct	1	CharacterString	

Attribute	primarySelector	Selection expression for instances of the first feature type	0..1	FeatureSelector	If omitted, all instances of the type are included. Example 1: categoryOfObstruction = 5 Example 2: waterLevelEffect = 4 AND WITHIN(<primary>, <secondary>)
Attribute	secondaryProduct	The other interoperating data product	1	dataProduct	
Attribute	secondaryFeatureCode	Feature type code in the FC for the product mentioned in secondaryProduct	1	CharacterString	
Attribute	secondarySelector	Selection expression for instances of the second feature type	0..1	FeatureSelector	If omitted, all instances of the type are included. Example 1: expositionOfSounding = 3 Example 2: categoryOfPile = 2 AND WITHIN(<primary>, <secondary>)
Attribute	outputProduct	Data product of the resulting hybrid feature	1	dataProduct	Default value = HYBRID, to indicate the result is a hybrid feature. This is a default – it can be set to other allowed values from the dictionary in specific cases.
Attribute	outputFeatureCode	Feature type code in the hybrid FC	1	CharacterString	
Role	featureRef	Reference to the output feature's display specification in the display planes section of the Interoperability Catalogue	1	<reference>S100_IC_Feature	

16-4.4.2.8 S100_IC_SuppressedFeatureInstance

S100_IC_SuppressedFeatureInstance is a class for suppressing feature instances according to attribute combinations in one product with features instances in another product. The primary product attribute combination specifies the feature instance(s) that will be replaced. The secondary product combination is the product to replace the suppressed instance(s) and should be identical to the output product attribute combination. If two different set of features instances are to be suppressed by one common set of feature instances, two instances of **S100_IC_SuppressedFeatureInstance** are needed.

Table 16-8 S100_IC_SuppressedFeatureInstance

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_SuppressedFeatureInstance	Operations for replacement of feature instances in one product by instances in another product.	--	--	Sub-class of S100_IC_FeatureDerivation The "secondary" product replaces the "primary".

Role	creationRule	Reference to a rule defined in the hybridization rules section of the catalogue	0..1	<reference>S100_IC_SimpleRule	

NOTE: If attributes bindings of the output are the same as secondary product type, the **outputProduct** and **outputFeatureCode** should be the same as the **secondaryProduct** and **secondaryFeatureCode.** If the attribute bindings change in a way that is incompatible with the Feature Catalogue for the secondary product, **S100_IC_HybridFeature** must be used instead.

16-4.4.2.9 S100_IC_HybridFeature

S100_IC_HybridFeature is a class for selecting primary and secondary inputs that will be suppressed from being rendered and replaced by a feature derived from the inputs.

Table 16-9 S100_IC_HybridFeature

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_HybridFeature	Class used to create a feature by combining feature types from two or more products for the purposes of an interoperable display	--	S100_IC_FeatureDerivation	Sub-class of S100_IC_FeatureDerivation
Role	creationRule	Reference to a rule defined in the hybridization rules section of the catalogue	0..1	<reference>S100_IC_CompleteRule	
Role	creationRule	Reference to a rule defined in the hybridization rules section of the catalogue	0..1	<reference>S100_IC_ThematicRule	

NOTE: Only one creationRule can be used in an instance.

16-4.4.2.10 S100_IC_HybridFeatureCreationRule

S100_IC_HybridFeatureCreationRule is an abstract super-class for different types of hybridization rules. *This functionality needs to be worked out but OGC Filter seems to be the ideal option for defining data filtering logic.* Overall, the output from execution of **S100_IC_HybridFeatureCreationRule** is a set of hybrid features for which predefined FC, PC and display plane definitions already exist so such feature will be suitable for passing to the portrayal engine for processing just like any other S-100 features. Instances of **S100_IC_HybridFeatureCreationRule** can be characterized by interoperability level, which allows the segmentation of different sets of interoperability operations depending on how tightly integrated the user desires the products to be on the resultant display, see 16-8 for more details.

Table 16-10 S100_IC_HybridFeatureCreationRule

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_HybridFeatureCreationRule	Class to capture hybrid feature creation rule captures the entire data filtering logic (that is, finding all features to be operated on) as well as the entire processing logic.	--	--	Abstract class

Attribute	interoperabilityLevel	The highest level of interoperability functionality encoded within an instance of this type	0..1	Integer	
Attribute	ruleIdentifier	Rule identifier	1	CharacterString	Mandatory unique ID used for references

16-4.4.2.11 S100_IC_CompleteRule

This class is used for rules describing how a set of feature instances is combined to create a hybrid feature type. Complete rules may operate on both thematic and spatial attributes.

Table 16-11 S100_IC_CompleteRule

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_CompleteRule	Class to capture rule describing how a set of feature instances is combined to create a hybrid feature type.	--	--	Subclass of S100_IC_HybridFeatureCreationRule

16-4.4.2.12 S100_IC_ThematicRule

This class is used for rules describing how a set of feature instances is combined to create a hybrid feature type. Thematic rules may operate on only thematic attributes. The input features are required to have spatially equal geometry within a tolerance set by the system.

Table 16-12 S100_IC_ThematicRule

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_ThematicRule	Rule describing how a set of feature instances is combined to create a hybrid feature type.	--	--	Subclass of S100_IC_HybridFeatureCreationRule

16-4.4.2.13 S100_IC_SimpleRule

This class is used for rules describing how a set of feature instances is combined to create a hybrid feature type. Simple rules may operate only on thematic attributes. Simple rules treat thematic attributes uniformly during hybridization and therefore do not mention specific attributes. Location/extent spatial attributes of all input features must be spatially equal.

Table 16-13 S100_IC_SimpleRule

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_SimpleRule	Rule describing how a set of feature instances is combined to create a hybrid feature type	--	--	Subclass of S100_IC_HybridFeatureCreationRule

16-4.4.2.14 S100_IC_HybridFC

Class for holding reference to a Feature Catalogue defining any "hybrid feature types" that are created by combining feature types from two or more products for the purposes of an interoperable display.

Table 16-14 S100_IC_HybridFC

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_HybridFC	Feature Catalogue defining any "hybrid feature types" that are created by combining feature types from two or more products for the purposes of an interoperable display.	--	S100_FC_FeatureCatalogue	The Interoperability Catalogue contains references to local resources (files) containing hybrid Feature Catalogues
Attributes and Roles	(See S-100 Part 5)				

16-4.4.2.15 S100_IC_HybridPC

Class for holding reference to a Portrayal Catalogue defining portrayal rules for the "hybrid feature types" defined in a Hybrid Features Catalogue.

Table 16-15 S100_IC_HybridPC

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_IC_HybridPC	Portrayal Catalogue defining portrayal rules for the "hybrid feature types" defined in a Hybrid Features Catalogue.	--	(S-100 Part 9) PortrayalCatalog	The Interoperability Catalogue contains references to local resources (files or folders) defining hybrid PCs
Attributes and Roles	(See S-100 Part 9)				

16-4.4.2.16 FeatureSelector

A data type for holding a template, logical expression, or match condition that, given a feature instance as parameter, can be evaluated to produce a TRUE/FALSE result.

Table 16-16 FeatureSelector

Type	Name	Description	Derivation	Remarks
datatype	FeatureSelector	A template, logical expression, or match condition that, given a feature instance as parameter, can be evaluated to produce a TRUE/FALSE result	subtype of CharacterString	Example: XSLT match condition

NOTE: Format and expression language must be defined in the implementation.

16-4.4.2.17 requirementType

A code list for sources of the Interoperability Catalogue or the person or party according to whose recommendations the catalogue was prepared.

Table 16-17 requirementType

Item	Name	Description	Code	Remarks
S100_Codelist	requirementType	The source of the catalogue or the person or party according to whose recommendations the catalogue was prepared	--	
Literal	IHO	Original IHO Interoperability Catalogue	1	
Literal	OEM	Prepared according to requirements specified by OEM or systems integrator	2	
Literal	national	Prepared according to requirements specified by a national government, group of national governments (for example the European Union), or governmental agency such as a national shipping authority or the Coast Guard.	3	
Literal	local	Prepared according to requirements specified by a sub-national governmental authority such as a state, province, or county	4	
Literal	port	Prepared according to requirements specified by a harbormaster's office or port authority	5	
Literal	company	Prepared according to requirements specified by the owner, charterer, or operator	6	
Literal	pilot	Prepared according to requirements specified by the vessel's master	7	
Literal	master	Prepared according to requirements specified by a pilot	8	
Literal	other	Other source	9	

Codelist Type: open enumeration

Encoding for extra values: other: <CharacterString> (Format of <CharacterString>: [a-zA-Z0-9]+([a-zA-Z0-9]+)* - See S-100 Part 3, clause 3-6.7).

16-4.4.2.18 dataProduct

A closed dictionary codelist of S-100 based products.

The allowed values are defined in the dictionary file, which is a component of individual interoperability specifications implementing this Part. If any operations or rules in an Interoperability Catalogue produce a hybrid feature, one of the entries in the dictionary must correspond to a notional “HYBRID” data product for such features.

Table 16-18 dataProduct

Item	Name	Description	Code	Remarks
S100_Codelist	dataProduct	List of data products	--	Data products conforming to the Specification identified by the item name, in the IHO list of S-100 based Product Specifications. This is a closed dictionary codelist (see S-100 1-4.8, 3-5.3.11, 3-6.7).
Literal(s)	(see individual specifications)	…	…	…

Codelist Type: closed dictionary, URI format: urn:mrn:…

URI tags must be fully specified in the implementation specification. E.g. urn:mrn:iho:prod:s98:1:0:0:products for a master list of products that is specified in S-98 as being covered by S-98 Version 1.0.0 Interoperability Catalogues.

The dictionary format used in Interoperability Catalogues is the ISO 19115-3 “codelist catalogue” format. An example using this format is depicted in Figure 16.4. The whole data products codelist is contained in the *CT_Codelist* XML element, identified by the XML ID “urn.mrn.iho.prod.s98.1.0.0.products” (the *id* attribute of *CT_Codelist*). Individual data products are listed in *codeEntry* elements within the *CT_Codelist*, identified by their own *id* attributes. In the example, S-101 is identified by the XML ID “s101” in the expanded *codeEntry* element. Note that the “:” characters in the codelist URI are replaced by “.” in the corresponding XML id elements, due to XML syntax constraints. The combination of *id* values of *CT_Codelist* and *CT_CodelistValue* elements is sufficient to identify a data product within a given dictionary file. The name (or other identifier) of the dictionary file must be specified in the individual interoperability specification. The Product Specification must also describe how dictionary files are made available to individual end user systems.

```
<cat:codelistItem>
 - <cat:CT_Codelist id="urn.mrn.iho.prod.s98.1.0.0.products">
     <cat:identifier>...</cat:identifier>
     <cat:name>...</cat:name>
     <cat:definition>...</cat:definition>
     <cat:description>...</cat:description>
   - <cat:codeEntry>
     - <cat:CT_CodelistValue id="s101">
       - <cat:identifier>
           <gco:ScopedName codeSpace="http://www.iho.int/S100/products">S-101</gco:ScopedName>
         </cat:identifier>
       - <cat:name>
           <gco:ScopedName codeSpace="http://www.iho.int/S100/products">S-101 ENC</gco:ScopedName>
         </cat:name>
       - <cat:definition>
           <gco:CharacterString>S-101 Electronic Navigation Charts</gco:CharacterString>
         </cat:definition>
       - <description>
           <gco:CharacterString>ENC data product</gco:CharacterString>
         </description>
       </cat:CT_CodelistValue>
     </cat:codeEntry>
     <cat:codeEntry>...</cat:codeEntry>
     <cat:codeEntry>...</cat:codeEntry>
     <cat:codeEntry>...</cat:codeEntry>
     <cat:codeEntry>...</cat:codeEntry>
     <cat:codeEntry>...</cat:codeEntry>
   </cat:CT_Codelist>
</cat:codelistItem>
```

Figure 16-4 - Example of data products dictionary entry

16-4.4.3 Filters

Attribute-value combination filters (the **attributeCombination** attribute of S100_IC_Feature) are strings of the form <attr><op><value>, where:

a) <attr> is the camel case code of the attribute;
b) <op> is one of "=", "!=", "in", "notIn", "gt", "ge", "lt", "le", "null";
c) <value> is a decimal number, integer, numeric code, or string, or a list of values. Strings must be enclosed in double quotes: "" with embedded double-quotes or \ characters preceded by a \ character.

The <attr>, <op>, and <value> components are separated by blank or tab characters[1].

Sub-attributes of complex attributes can be indicated in <attr> fields using a restricted subset of relative path expressions as specified in the W3C XPath specification (§ 3.3.1 XML Path Language (XPath) 3.1). The restrictions are:

- Paths are relative to the individual feature as the context node.
- Only the "child" axis is permitted and the optional "child::" prefix is not used.
- Predicates as described in the XPath specification are not used.

The effect is to allow <attr> fields to describe sub-attributes in terms of camel case codes separated by "/" characters. (It also allows a simple attribute to be designated by its camel case code alone as described above.)

EXAMPLE 1: An **attributeCombination** with value *categoryOfRadioStation = 20* selects features with **categoryOfRadioStation** attributes that have the value 20 (AIS Base station).

EXAMPLE 2: An **attributeCombination** with value *featureName/language = "eng"* selects features with **featureName** attributes that have a language sub-attribute having the value "eng".

> Note (informative): Selectors may relax these restrictions and use a larger subset of XPath. Details should be elaborated in the implementation specification.

16-5 Correspondence to and suppression of Portrayal Catalogue elements

Correspondences between Interoperability Catalogue and Portrayal Catalogue elements are summarized in Table 16-19 below. In all cases, the supersession of Portrayal Catalogue display by Interoperability Catalogue display applies only to the subset of features remaining after applying the filter described by the attributes geometryType and attributeCombination.

Table 16-19 - Correspondences between display instruction elements in Interoperability Catalogues and Portrayal Catalogues

Interoperability Catalogue element	Portrayal Catalogue element	Interoperability Catalogue supersedes Portrayal Catalogue?	Note
product	(implicit) aggregation in DisplayList	Not applicable	Implicit in aggregation in DisplayList; the product can be identified from data product to which the Portrayal Catalogue as a whole applies.

[1] More expressive filter expressions can be developed if required for advanced interoperability.

drawingPriority (S100_IC_DrawingInstruction) drawingPriority (S100_IC_Feature)	drawingPriority	Y	
identifier	--	Not applicable	
viewingGroup	viewingGroup	Y	
geometryType	(implicit)	Y	Can be derived from the implementing element in the Portrayal Catalogue, for example PointInstruction, AreaInstruction, LineInstruction.
attributeCombination	(XSLT template)	Y	XSLT template in Portrayal Catalogue
featureCode	(implicit in XSLT template)	Not applicable (must be compatible by definition)	Interoperability Catalogue elements pertain to feature types or subsets of feature types; instances of drawing instructions in Portrayal Catalogues reference individual features.
(composition association)	displayPlane	Y	Composition to S100_IC_DisplayPlane.
--	scaleMinimum, scaleMaximum	Not applicable	
--	featureReference association	Y	Interoperability Catalogue elements do not refer to individual instances (see Example 1).
--	spatialReference association	Y	Interoperability Catalogue elements do not refer to individual instances (see Example 2).
substituteSymbolization (S100_IC_DrawingInstruction only)	(symbolization generated by portrayal processing)	Y	(See Example 3).

EXAMPLE 1: A Portrayal Catalogue for an overlay product places all its features in the same display plane, but the Interoperability Catalogue splits them into over-radar and under-radar planes.

EXAMPLE 2: Feature **Current – Non-Gravitational** is allowed to have point, curve, or surface geometry. It is possible for a Portrayal Catalogue to place them in different viewing groups depending on the type of spatial primitive and for an Interoperability Catalogue to override that placement and put all instances of this feature, whether point, curve, or surface, in the same viewing group.

EXAMPLE 3: The area boundary symbolization is changed from a simple to a composite line style for area boundaries common to different types of area features, for example an anchorage area in product A bordering a marine protected area in product B.

16-6 Feature creation or replacement rule

Feature creation and replacement rules create new feature types by combining characteristics of specified feature types from the input data products. A feature creation/replacement rule basically transforms a collection of feature instances in the input stream into one or more different feature instances in the output stream. The created feature differs from all the input features, for example by adding properties of one feature to properties of another feature.

Since new feature types must be defined in a Feature Catalogue for ECDIS use, there is also a “hybrid” Feature Catalogue that contains feature type specifications for all the possible feature types which can be thus generated by rules in an Interoperability Catalogue. Similarly, the hybrid Portrayal Catalogue describes the portrayal of these feature types.

The process for applying such rules is illustrated in Figure 16-5 below.

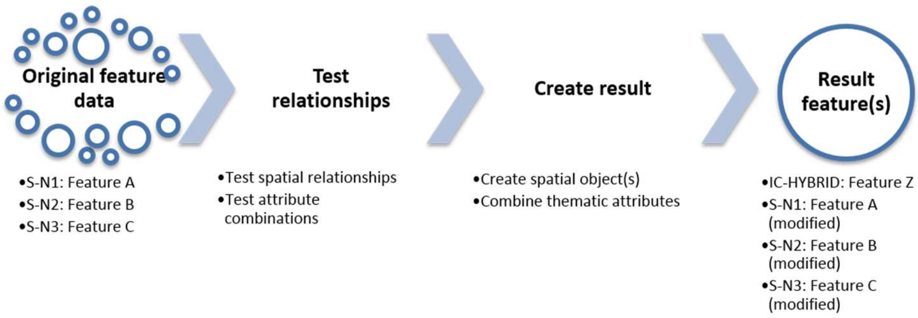


Figure 16-5 - General process for feature creation and replacement rules

A hypothetical example of the operation of such a rule is depicted in Figure 16-6 below. On the left are two hypothetical input features:

- Feature instance DRGARE_12345 from a chart layer, an instance of feature type DredgedArea. Its geometry is an area, depicted alongside.
- Feature instance AISMessage_8472 from a met/hydro layer, an instance of feature type MetHydroDataAISMessage. Its geometry is a point, depicted alongside.

The operation of the rule results in two feature instances, described in Table 16-20 below:

Table 16-20 - Features resulting from operation of hypothetical rule

Output feature instance	Output feature Type	Defined in Feature Catalogue	Spatial attributes	Thematic attributes
RTWL_H01	RealTimeWaterLevelArea	Hybrid FC IHOICFC01.XML	Circle centred at location of input AISMessage_8472	Combination of DRGARE_12345 and AISMessage_8472
DRGARE_H01	DredgedArea	Chart FC	Spatial difference of original DRGARE_12345 and circle geometry of RTWLArea_H01	Same as DRGARE_12345

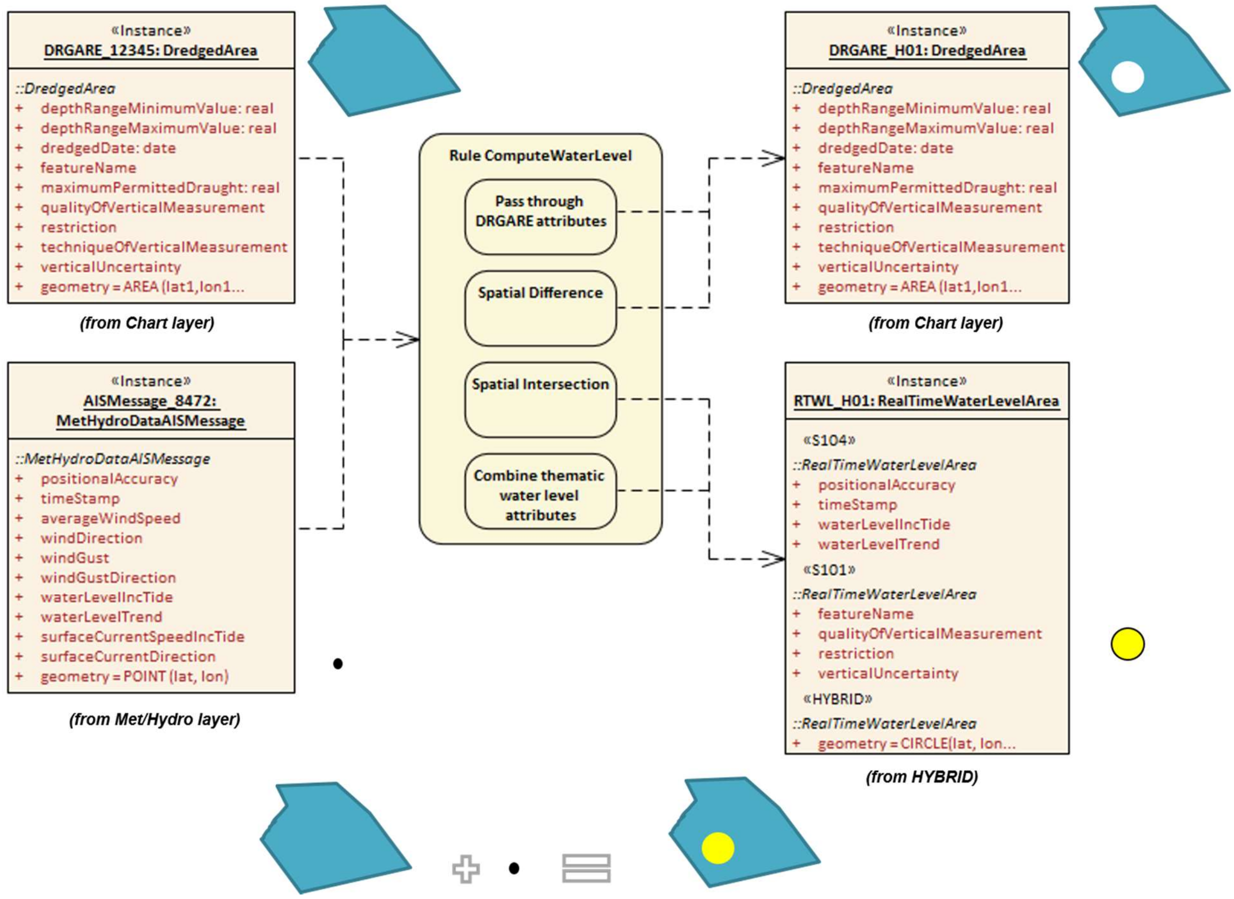


Figure 16-6 - Illustrative example of a creation/replacement rule

16-7 Interoperability Catalogue Data Model and the General Feature Model

This clause elaborates on how General Feature Model (Part 3) elements are managed by the Interoperability Catalogue Data Model.

16-7.1 Geographic feature types

The relative prioritization for display purposes of geographic feature types is at the core of the Interoperability Catalogue Data Model. For the Interoperability Catalogue concept, feature types can be considered as the “domain” of the Data Model, just as feature concepts form the domain of an ordinary Product Specification.

Feature instances are not encoded in Interoperability Catalogues since an Interoperability Catalogue is a catalogue-based product that is functionally a collection of rules which adjust the display of information from feature datasets; an Interoperability Catalogue is not itself a feature-based data product.

References to feature types may appear as attribute values in Interoperability Catalogues. The reference will identify the Product Specification in which the feature type is defined. It may also identify the version of the Product Specification; if the version is not identified the reference is to the indicated feature type in all versions of the Product Specification.

A reference to a feature type must be interpreted as applying to all instances of the feature type in datasets conforming to the indicated Product Specification and version. (Additional conditions limiting applicability to subsets of feature instances may be encoded in other attributes.)

16-7.2 Meta feature types

The suppression, interleaving, and replacement operations in the Interoperability Catalogue do not affect meta features in individual Product Specifications. Display of meta features if requested by the mariner is as specified by individual Portrayal Catalogues.

16-7.3 Feature and Information associations

Feature and information associations are not directly used in an Interoperability Catalogue.

16-7.4 Information types

Information types are not directly used in an Interoperability Catalogue.

16-8 Interoperability Levels

The **interoperabilityLevel** attribute (**S100_IC_InteroperabilityCatalogue, S100_IC_DisplayPlane, S100_IC_PredefinedCombination, *S-100_IC_HybridFeatureCreationRule***) is used to indicate a level of interoperability. The value 0 means interoperability is off and all layers are portrayed according to their stacking order. When a value of 1 or higher is assigned to the interoperabilityLevel attribute it means the interoperability functionality is on. The exact meaning of any value of 1 or higher must be described in the Interoperability Catalogue specification where model is implemented.

Different product combinations may require different types of interoperability in order to work as intended in a system. For example, a product combination may require the ability to turn off a feature in product A to give prominence to a feature in product B. Another example may be that a product combination requires the ability to place product C in-between layers within product D. Within the same Interoperability Catalogue these different types of interoperability can be distinguished using different values in the **interoperabilityLevel** attribute of the classes that make up the functions.

Example: a specification has defined four levels of interoperability in addition to the so-called Level 0. The four levels start out with simple interleaving at Level One to virtually combing data at Level Four as seen in the example Figure 16-7 below.

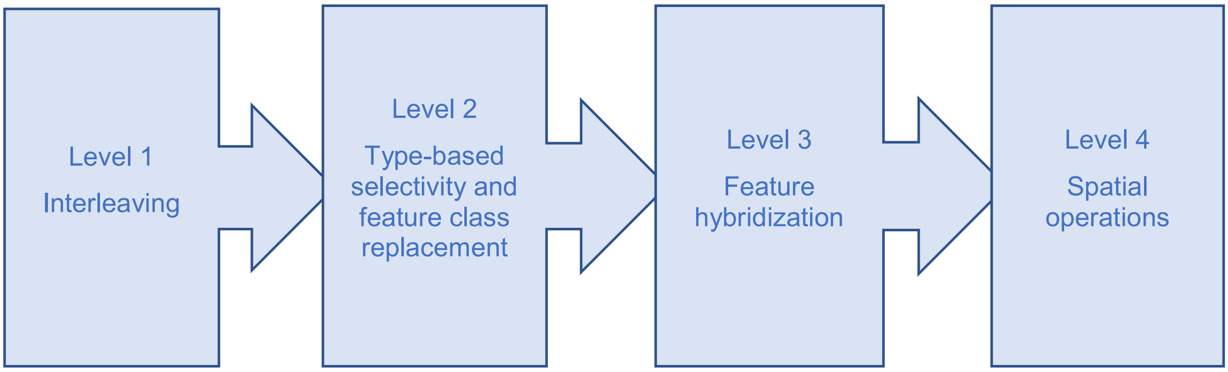


Figure 16-7 - Example of how to separate interoperability functions into levels

16-9 Hybridization rules

Hybridization rules define how a set of feature instances are combined to create a hybrid feature type. In the simplest form of hybridization, and assuming that there are no collisions between the thematic attributes of the input types, the hybrid feature would bind all the attributes of the input types to a single output feature type. More complex hybridization rules can handle collisions, for example by defining a preference order for colliding attributes, including all the values if the input types bind the same enumerated attribute, or adding uncertainty metadata if numeric attribute values are different. The hybridization rules require two feature instances as input and produce a single feature instance as output.

> Note: The formal specification and rule language for hybridization will be described in a subsequent version of this document.

16-9.1 Simple hybridization rule

Simple rules treat thematic attributes uniformly, for example by binding the attributes of both primary and secondary input instances to the output instance, or preferring the attribute bindings of the primary instance to those of the secondary instance in case of a difference in the values of common attributes. Location/extent spatial attributes of all input instances must be spatially equal and are passed through unchanged.

16-9.2 Thematic hybridization rule

Thematic rules treat thematic attributes on an individual basis, for example, use specified attributes from the primary input instance and specified attributes from the secondary input instance. Combination operations on attribute values may be specified, for example, OutputFeature.depthValue = maximum(ProductA.FeatureX.depthValue, ProductB.FeatureY.depthValue. Location/extent spatial attributes of all input instances must be spatially equal and are passed through unchanged.

16-9.3 Complete hybridization rule

Complete rules allow selection of input sets using complex spatial queries as well as spatial equality and selector expressions on attribute values. The output can combine thematic attributes in any of the ways allowed by thematic hybridization rules. In addition, it may generate complex spatial objects from the input spatial primitives by applying selected spatial operations to the input instances. The allowed spatial operations will be identified in a future edition of this specification (tentatively, the spatial operations defined in IHO S-58, Section 2).

16-9.4 Hybrid Feature and Portrayal Catalogues

Hybrid Feature and Portrayal Catalogues are physically separate files from the main Interoperability Catalogue, but the main Catalogue links to them by encoding the names of the hybrid catalogue files which are used by the feature creation rules defined in it. The hybrid Feature and Portrayal Catalogues conform to the structures required by S-100 Parts 5 and 9 respectively. It is recommended that hybrid Feature and Portrayal Catalogues be bundled with the Interoperability Catalogue in an exchange set as support files, see Clause 16-10 below.

16-10 Production process of Interoperability Catalogues

Interoperability Catalogues are created in an XML editor environment. It is generally expected that any off-the-shelf XML Editor can perform this task. The creation process of any new versions may benefit from starting from the previous version when available. Another alternative is to develop a dedicated Interoperability Catalogue Editor that can create and maintain Interoperability Catalogues with dedicated functions for creating the various components, such as display planes and predefined combinations.

Due to the interconnected nature of the Product Specifications that are under the Interoperability Schema, a form of overarching change management is a necessity. Any revision or New Edition required in a Product Specification should be announced well in advance, giving the whole stakeholder community ample time to review the impact before it goes into effect. Any revisions and New Editions to a supported Product Specification may require a new version of an Interoperability Catalogue and the party responsible for the maintenance of an Interoperability Catalogue needs to be informed and involved to assess any impacts. This includes updates to dataset metadata, as metadata changes such as Product Specification references may impact the link between the dataset and the Interoperability Catalogue.

It is recommended that Interoperability Catalogues be distributed in an exchange set structure, see Part 17. Discovery metadata for Interoperability Catalogues must be provided in XML format conforming to S-100 dataset discovery as specified in S-100 Part 17, with the extensions described in this clause.

16-10.1 Management of Feature Catalogue and Portrayal Catalogues updates

Changes to a supported Product Specification may have impacts on an Interoperability Catalogue. Revisions to the Feature Catalogue or Portrayal Catalogue are the most likely to require a revision of the Interoperability Catalogue in order to support the change. These types of changes will generally require a new version (n.n.0) of the Interoperability Catalogue to ensure support. It should be noted that revisions to a supported Feature Catalogue or Portrayal Catalogue may be ignored by previous versions of an Interoperability Catalogue and it is therefore necessary to consider this as part of the change management process, especially if the change is a matter of navigational safety. Major changes to Product Specifications, such as adding functionality or adding new Product Specifications to the supported list will result in a new edition (n.0.0) of an Interoperability Catalogue.

Versions of an Interoperability Catalogue within the same edition are considered a compatible group. When a New Edition is issued, this compatibility is broken; and efforts should be undertaken to update all impacted systems as soon as possible.

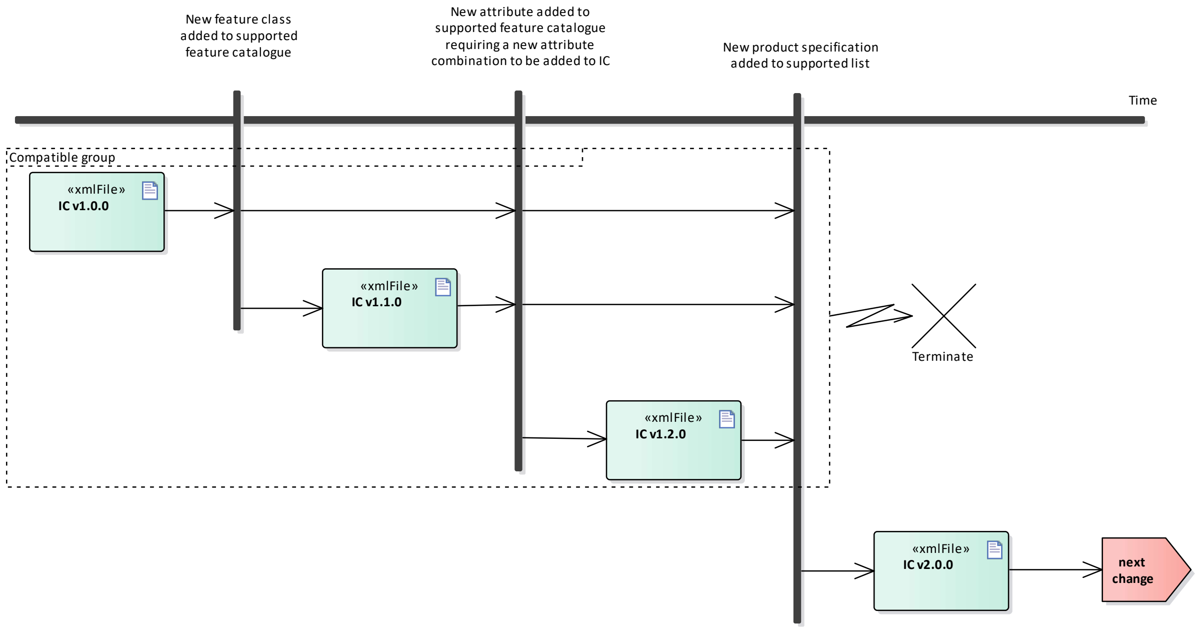


Figure 16-8 - Examples of how Feature Catalogue change may impact Interoperability Catalogue lifecycle

Several types of changes to supported Product Specifications may impact an Interoperability Catalogue in such a way that a new version is needed to maintain full support. These include:

- New feature added to a supported Product Specification that requires a new feature combination to be added to an Interoperability Catalogue;
- New attribute added to a feature in a supported Product Specification that requires a new attribute combination to be added to an Interoperability Catalogue;
- New Product Specification is added to the list of supported Product Specifications in an Interoperability Catalogue. This could also require new feature and attribute combinations to be added;

- Removal of feature or attribute from a supported Product Specification that are present in a feature or attribute combination within an Interoperability Catalogue;
- A correction to a supported Product Specification that triggers a version increment (n.**n**.0), which may break links from an Interoperability Catalogue to the supported Product Specification;
- Matters of navigational safety as they arise.

Adding new functions in either a supported Product Specification or an Interoperability Specification may require a new version of an Interoperability Catalogue.

16-10.2 Product Specification updates other than FC/PC

Updates to dataset metadata, such as Product Specification references, may impact the link between a dataset and an Interoperability Catalogue and therefore require a revision to the Interoperability Catalogue (n.**n**.0). Some changes to an interoperability-ready Product Specification may not require any changes to an Interoperability Specification or an Interoperability Catalogue. This includes amendments to the definitions of features, attributes or attribute values. It also includes minor changes to Product Specifications, such as clarifying language. Other changes may only require an update to the metadata of an Interoperability Specification and/or Interoperability Catalogue, such as in case of supported Product Specification version references. Such minor changes to an Interoperability Specification and/or Interoperability Catalogue may be collected and be applied at a later time when a more substantial revision is required.

16-10.3 Unpredictable Product Specification updates

Unpredictable changes to an interoperability-ready Product Specification, or its Feature Catalogues and Portrayal Catalogues, should be avoided. Great care should be taken in coordinating changes among all stakeholders to avoid any unforeseen consequences. Product Specifications that are under the same interoperability umbrella should be considered interconnected. The responsible groups should therefore coordinate changes with other groups that issue interoperability-ready Product Specifications. Such coordination can be done, for example, by having an agreed fixed period between releases in which all pending changes are collected, implemented and issued in a coordinated fashion.

If a situation arises where a product is not compatible with an Interoperability Catalogue, this product will only be available as an overlay that may obscure any data below the incompatible product, or the overlay may be obscured by any data with higher priority.

16-10.4 Additional data products to be defined in the Catalogue

An Interoperability Catalogue should in theory be extensible to products not yet defined, provided these are within the same S-100 Edition. New products can be included in a Catalogue, but such scenarios will require a New Edition of an Interoperability Catalogue. The default processing for any Interoperability Catalogue is that any new product follows the default rule of being used as an overlay, generally stacked in a user defined order and with default viewing groups in effect.

16-10.5 Backward compatibility

Different versions of data products may be simultaneously active; the Interoperability Catalogue design allows for backward-compatible updates if and when the Interoperability Catalogue has to be updated, within the same major Edition. Figure 16-9 below shows an example of how the Interoperability Catalogue may evolve with change over time.

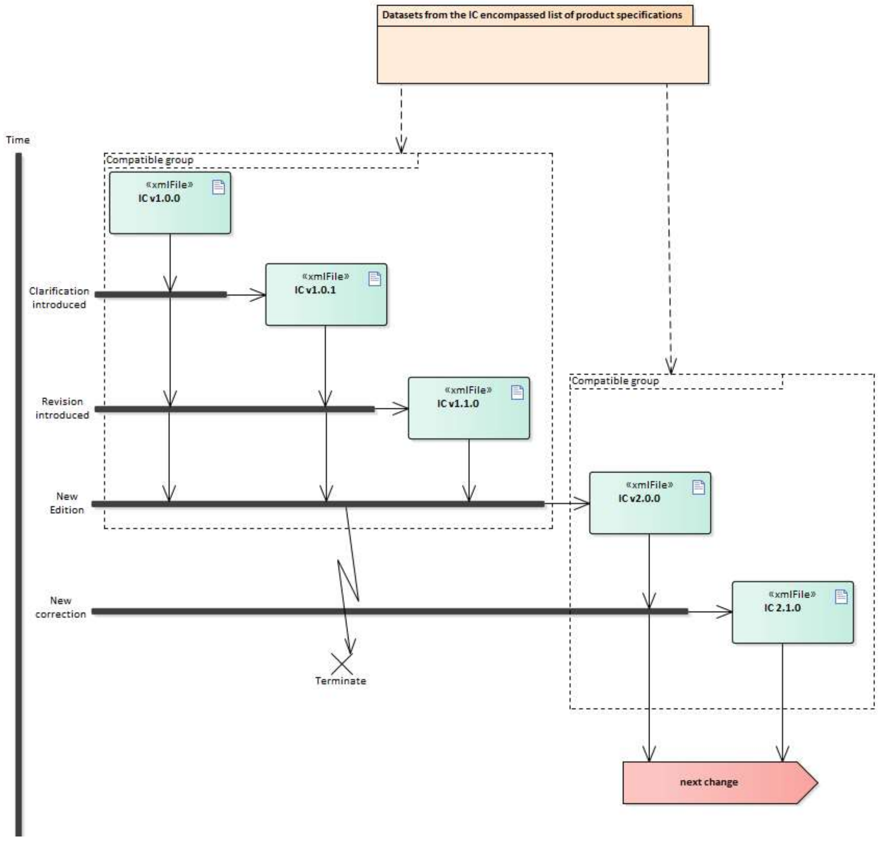


Figure 16-9 - Example life cycle of Interoperability Catalogue

16-10.6 Interoperability Catalogue product

Each Interoperability Catalogue product should by itself be considered a whole unit. New versions – either clarification, correction, or New Edition – should be updated by replacement with a newer version. The update process should be specified in detail when defining an Interoperability Specification. The following clauses give some principles to consider for the specific update process.

16-10.6.1 Updating the Interoperability Catalogue

It is recommended that new versions of the Interoperability Catalogues be distributed using an Exchange Set structure, see Part 17.

Interoperability Catalogues may remain active after a new version has been issued. Systems receiving new versions within the same major edition should retain all versions. Store these in separate folders to avoid any issues, such as when the same support files have been reused between versions.

Due to issues with broken backwards compatibility, all previous versions of the Interoperability Catalogue should be cancelled when a New Edition is issued.

16-10.6.2 Cancelling a version of the Interoperability Catalogue

In order to cancel a version of the Interoperability Catalogue it is recommended that the methodology used for exchange set as defined in Part 17 metadata is utilized.

16-10.7 Support files

Support files to an Interoperability Catalogue are generally the hybrid Feature Catalogues and hybrid Portrayal Catalogues. Methods for managing these are described below.

16-10.7.1 Updating the Interoperability Catalogue support files

It is recommended that support files are updated using the methodology used for exchange set as defined in Part 17 metadata is utilized.

Support files should be stored in a separate folder within the exchange set.

16-10.7.1.1 New Edition of the support files

New Editions of the support files introduce significant changes. New Editions enable new concepts, such as the ability to support new functions, or the introduction of new constructs. New Editions are likely to have a significant impact on either existing users or future users of an Interoperability Catalogue Specification.

EXAMPLE: A new product is added to Interoperability Catalogue, and all support files should be updated to support the new product. This would require a New Edition of the support files.

16-10.7.1.2 Revisions of the support files

Revisions are defined as substantive semantic changes to the support files. Typically, revisions will change the support file to correct factual errors; introduce necessary changes that have become evident as a result of practical experience or changing circumstances. A revision must not be classified as a clarification. Revisions could have an impact on either existing users or future users of an Interoperability Catalogue Specification. All cumulative clarifications must be included with the release of approved revisions.

Changes in a revision are minor and ensure backward compatibility with the previous versions within the same Edition. Newer revisions, for example, introduce new feature or attribute combinations. Within the same Edition, a support file created for an Interoperability Catalogue of one version could always be processed with a later revision.

EXAMPLE: Adding a new hybrid feature will require a revision increment to the support file.

16-10.7.1.3 Clarifications to the support files

Clarifications are non-substantive changes to the support file. Typically, clarifications: remove ambiguity; correct grammatical and spelling errors; amend or update cross references; insert improved graphics in spelling, punctuation and grammar. A clarification must not cause any substantive semantic change to an Interoperability Catalogue Specification.

Changes in a clarification are minor and ensure backward compatibility with the previous versions within the same Edition. Within the same Edition, a support file created for an Interoperability Catalogue of one version could always be processed with a later clarification (or revision) of an Interoperability Catalogue.

EXAMPLE: Correcting a spelling error in a definition will require a clarification increment to the support file.

16-11 Portrayal

This clause gives guidelines and instruction to portrayal considerations related to the use of the Interoperability Catalogue in a user system. An Interoperability Catalogue must apply to the specific Product Specifications listed in its metadata, specifically the **productCovered** attribute under **S100_IC_InteroperabilityCatalogue**.

There may be additional data products present in the user system that are external to the Interoperability Catalogue; in such cases the Interoperability Catalogue should continue to function in presence of product not defined in the Catalogue. Data products that are outside of the interoperability scope must be treated as overlays (see clause 16-10.4).

16-11.1 Display of significant features

Significant features in a display plane should have the highest drawing priority value within the **S100_IC_DisplayPlane**. Care should also be given to assigning significant features with high drawing priority values within the relevant **S100_IC_Feature** and **S100_IC_DrawingInstruction**. This ensures that less significant features in one data product are not displayed more prominently than more significant features in another product.

16-11.1.1 Switching to original

User systems should include means to easily switch on and off the interoperability functionality.

16-11.2 Impacts on viewing groups

The viewing group is a concept to control the content of the display. It works as an on/off switch for any drawing instruction assigned to the corresponding viewing group. The concept can be seen as a filter on the list of drawing instructions [S-100 Part 9, clause 9-11.1.3].

Viewing groups in Interoperability Catalogues take precedence over the applicable viewing groups for any feature instances from a supported Product Specification, and that are included in a **S100_IC_DrawingInstruction**, or **S100_IC_Feature** instance.

16-11.3 Impacts on Portrayal Catalogues

Viewing systems must manage the visibility and drawing order of data products, especially relative to radar/ARPA, AIS display or other sensor data. Moreover, systems must ensure significant features with over radar flag, in all products, are distinguishable in the presence of radar/ARPA, AIS or other sensor data. In navigation systems tracks and vessel position information are likely high priority while some AIS ASM (application specific messages) may carry lower-priority information, including data described by an S-100 based Product Specification, for example meteorological and hydrographic information.

16-11.4 Meta-features

It is recommended that viewing systems allow display of meta features for only one product at a time. This is in order to minimize risks of display clutter, user confusion, and the possibility of interpreting meta-features for one product as applying to a different product.

For example, data quality meta features for different on-screen products should not be displayed simultaneously, and that only the top most product data quality should be shown at any given time. This also applies in areas of the screen where the topmost product does not cover.

16-11.4.1 Data quality for individual products

This clause applies to the case where multiple products are on-screen and quality meta-features are enabled. Only one set of quality features should be displayed at any given time to avoid clutter and misreading the meaning of the quality metadata.

Interoperability Catalogues do not specify means of distinguishing data quality portrayals for individual products. Product Specifications must provide rules for display of data quality metadata (including data quality meta-feature information), which the user system will utilise to portray data quality.

Means of distinguishing data quality portrayals for individual products is left to the Product Specification authors (in particular, Portrayal Catalogue authors) and OEMs, and can be handled by distinguishing portrayal rules or symbology for different products' data quality meta-features, such as colour coding or special line symbol. There should also be a clear on-screen message saying what data quality features are displayed in order to give users a firm indication of the layer to which the currently displayed quality metadata applies.

16-11.4.2 Portrayal of data quality for combinations

Interoperability Catalogues do not include combining data quality portrayals. The recommendation in clause 16-11.4 about displaying only one set of meta-features is strengthened for data quality in particular. It is recommended that simultaneous portrayal of data quality from different products be avoided.

It is therefore recommended that user systems include functions to let the user select which product's data quality should be displayed.

16-11.5 Display of text

Text is typically the last item drawn. In general, rules for placement, display selection, and management of long text are defined in the individual Product Specification and associated Portrayal Catalogue. An Interoperability Catalogue would, in general, only govern when a feature that text is generated from is displayed.

Developers of user systems with interoperability capabilities should consider adding functions for enhanced automatic text placement to limit clutter.

16-11.6 Blended feature concepts or blended portrayals

This clause describes the interoperability solution for blended feature or blended portrayal. These can be produced by using transparency or creating a temporary blended feature or blended portrayal (rule and/or symbol) of specific combinations of features from different products. Such blended concepts will typically be created by using **S100_IC_PredefinedCombination** which links to a hybrid Portrayal Catalogue that includes the features to be combined and a suppression rule (using **S100_IC_SuppressedFeatureLayer**) for the features that are to be replaced.

An example where a blended concept could be used is where winds blowing from the west cause fairways to some west coast ports of Finland to get layered ice (wind pushes ice layers on top of each other until there is ice from the sea bottom up to the surface). When an ice-breaker makes a path through some ice remains between the sea bottom and the keel of the ice-breaker. Ice thickness in such a place could be up to 11 m while the ice-breaker draught is around 7-8 meters. In such cases a simultaneous display of both ice coverage and underlying depth area is required. Other depth area features such as spot soundings, rocks, wrecks, etc., are also still important.

16-11.7 Blended portrayal's effect on symbols and area patterns

When combining various layers, that may be of different compilation scale and coverage, it is likely that symbols and area patterns will end up at borders, or conflict with symbols and area patterns in other layers. It is important that symbols remain legible, and that user systems include appropriate methods to avoid displaying partial symbols, or "grafting" part of lower-layer symbols onto symbols in upper layers.

For area fills the symbols of a pattern fill must be closer together for a small or narrow area, to ensure enough symbols are seen, and farther apart for a large area, to avoid clutter. An area pattern may be substituted by a single centred symbol if sufficient space is not available for a pattern to be shown.

16-11.8 Hierarchy of data

Hierarchy between different Product Specifications is influenced by several factors such as intended use within the in-scope user system, for example usefulness a product in navigational operation in a navigation system. It may not be possible to prescribe a fixed hierarchy list as a universal standard, and the Interoperability Catalogue model therefore offers a flexible approach. Within the Interoperability Catalogue the hierarchy of data between different S-100 based Product Specifications is determined using predefined combinations. The **S100_IC_DisplayPlanes** referenced within a **S100_IC_PreDefinedCombination** give the order of feature layers. This approach also allows for different stacking orders of the same products within the same Interoperability Catalogue should there be a user need for this. Care should be used (for example by using descriptive names and allowing users to see these when selecting) to ensure the risk of users unintentionally selecting the wrong stacking order.

16-11.9 Interacting gridded information

If two or more gridded data types are to interact, the hierarchy between them should be established using predefined combinations as with other data types. Particular care has to be taken depending on how the presentation of the data is to be done when deciding which gridded data type has the highest priority, considering items such as will one gridded data type obscure the other.

For example, gridded bathymetry may obscure gridded surface currents and therefore the gridded surface currents should be given the highest priority between the two when they are to be displayed simultaneously.

16-11.10 Pick reports

Pick reports may be defined in the individual Product Specification. The Interoperability Catalogue model permits reuse of these specifications as it does not specify pick report design for the individual supported Product Specification.

Complete data from all products visible on the screen should be available to the system user, irrespective of all these products being in the scope of the Interoperability Catalogue in use or not.

Features that have been visually suppressed should not be included in the pick report.

16-11.11 User control over loaded set

It is recommended that users have the option to load additional products in scope for the system, even when these are out of scope for the Interoperability Catalogue, or turn off one or more of the data products in a predefined combination. Portrayal must adjust to the loaded set as appropriate, for example if an additional product is loaded, it should be interleaved with layers from data products in the predefined combination according to the drawing order and drawing priority in its Portrayal Catalogue.

16-11.12 User control over interoperation level

If more than one interoperability level is supported by the Interoperability Catalogue (see 16-8), it is recommended that users have the option to select the interoperability levels they wish to use. Portrayal must adjust to the new interoperation level.

Page intentionally left blank

Appendix 16-A
Interoperability Catalogue XML Schemas
(normative)

16-A-1 XML Schema

16-A-1.1 Overview

Developers of S-100 Interoperability Catalogues must develop their data format consist with the Interoperability Catalogue XML Schemas referenced in this Appendix. The Schemas have a root or container element **S100_IC_InteroperabilityCatalogue**, whose structure is shown in Figure 16-A-1 below.

16-A-1.2 Unknown attribute values

When a mandatory attribute code or tag is present but the attribute value is missing, it means that the producer wishes to indicate that this attribute value is unknown. Missing mandatory attributes must be "nilled". Optional attributes must be omitted altogether if the value is unknown or missing. They must not be "nilled". This rule also applies to metadata.

16-A-2 Interoperability Catalogue structure

An Interoperability Catalogue contains header information identifying the catalogue (a specialization from **CT_Catalogue** and extended with specific elements appropriate to S-100 interoperability, such as digital signatures and elements identifying the source of interoperability requirements). The header is followed by a list of products covered by the catalogue and containers for display planes, predefined combinations, hybridization rules and references to the hybrid portrayal and Feature Catalogues used by the results of hybridization rules. The general structure is depicted in Figure 16-A-1.

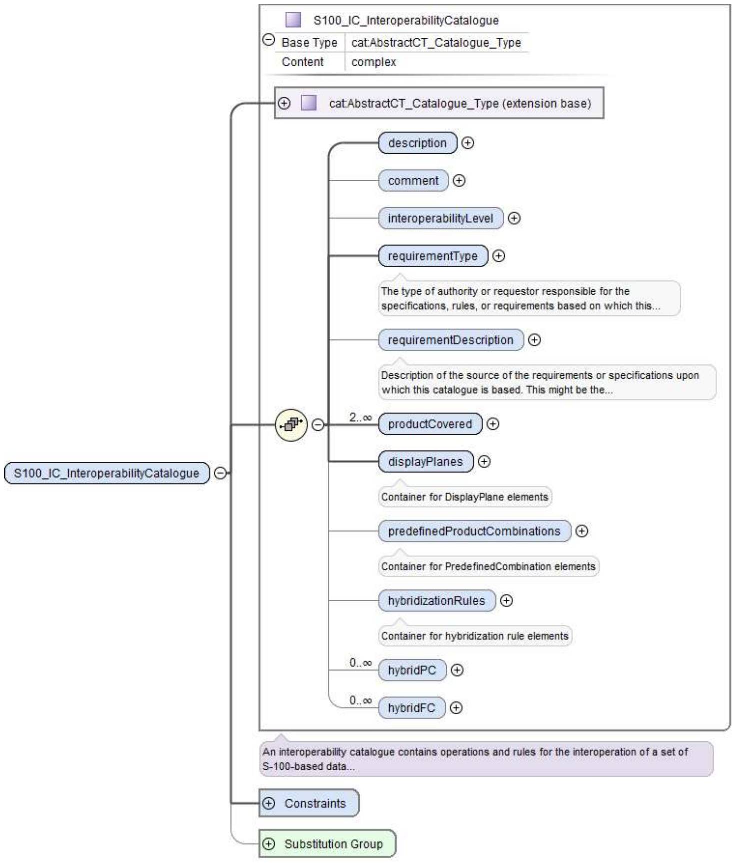


Figure 16-A-1 - Basic structure of an Interoperability Catalogue

The *displayPlanes* container is a collection of 1 or more **S100_IC_DisplayPlane** elements. The *predefinedProductCombinations* container is a collection of 0 or more **S100_IC_PredefinedCombination** containers. (The difference in the lower bound arises from the fact that it is possible for a catalogue to implement only interleaving but no higher level of interoperability.) The containers are depicted in Figure 16-A-2 and Figure 16-A-3 below.

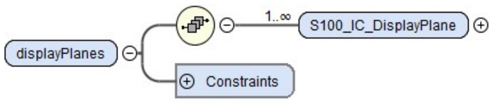


Figure 16-A-2 - Container for display planes

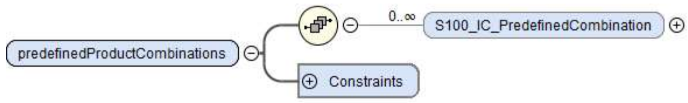


Figure 16-A-3 - Container for predefined product combinations

The elements in each **S100_IC_DisplayPlane** and **S100_IC_PredefinedCombination** correspond to the model described in clause 16-4 and are depicted in Figures 16-A-4 and 16-A-5 below. The contents of these elements are:

- **S100_IC_DisplayPlane** element:
  - the interoperability level;
  - the priority for the plane (displayPriority);
  - identifying and descriptive elements (identifier, name, description);
  - containers for **S100_IC_Feature** and **S100_IC_DrawingInstruction** elements.
- **S100_PredefinedCombination** element:
  - the interoperability level;
  - a list of the data products covered by this predefined combination (includedProduct elements);
  - references to **S100_IC_DisplayPlane** elements;
  - optional containers for feature suppression rules and feature derivation rules;
  - identifying and descriptive elements (identifier, name, description, useConditions).

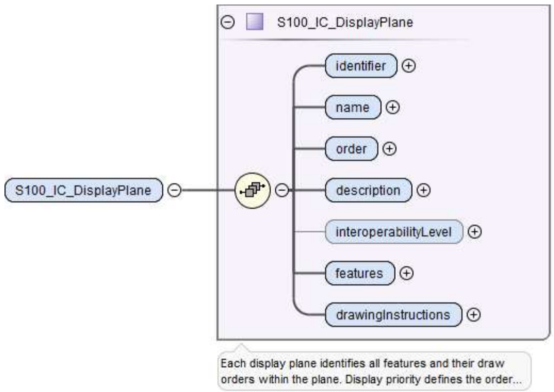


Figure 16-A-4 - Structure of individual display plane element

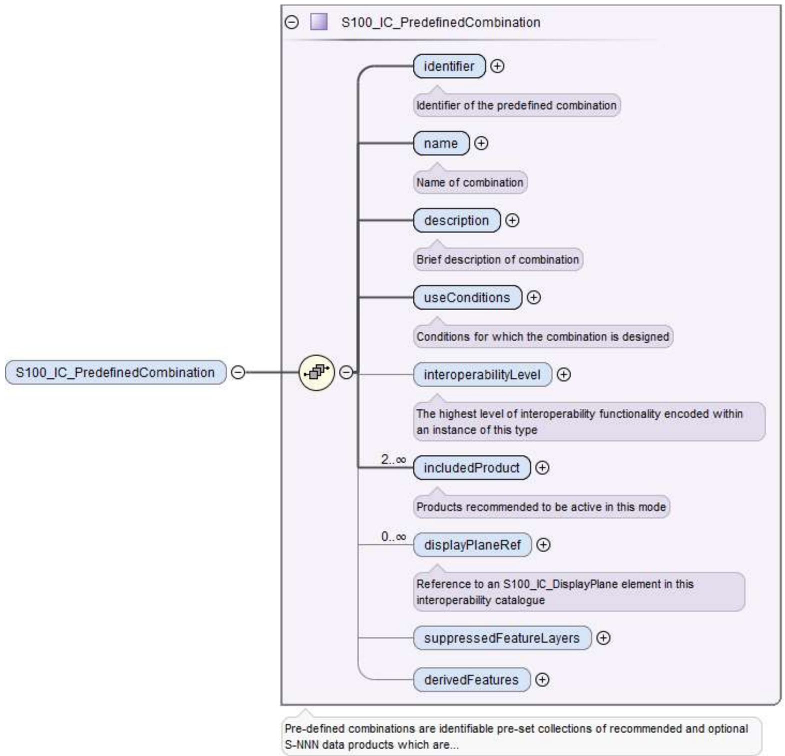


Figure 16-A-5 - Structure of individual predefined combination element

The structures of **S100_IC_Feature**, **S100_DrawingInstruction**, and **S100_IC_SuppressedFeatureLayer** elements correspond to the model and documentation in clause 16-4 and are depicted in Figure 16-A-6, Figure 16-A-7 and Figure 16-A-8 below.

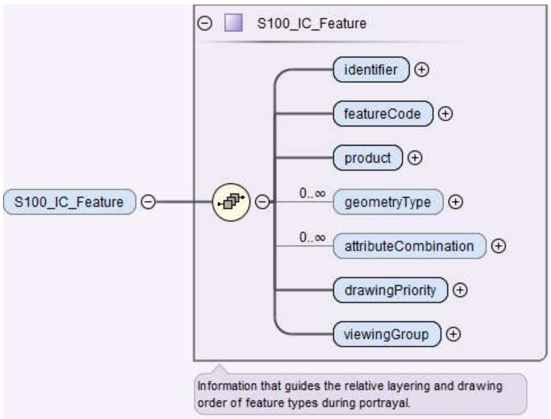


Figure 16-A-6 - Structure of S100_IC_Feature

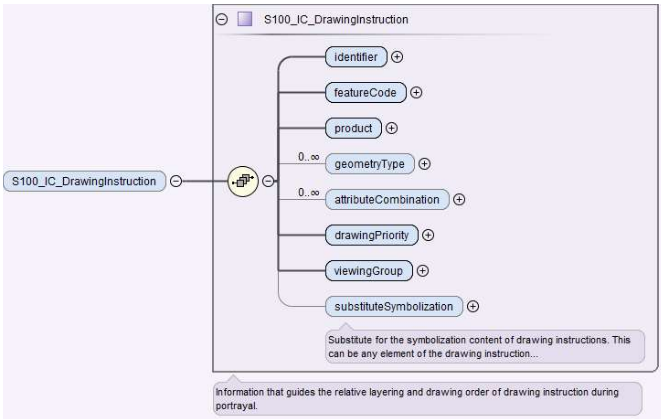


Figure 16-A-7 - Structure of S100_IC_DrawingInstruction

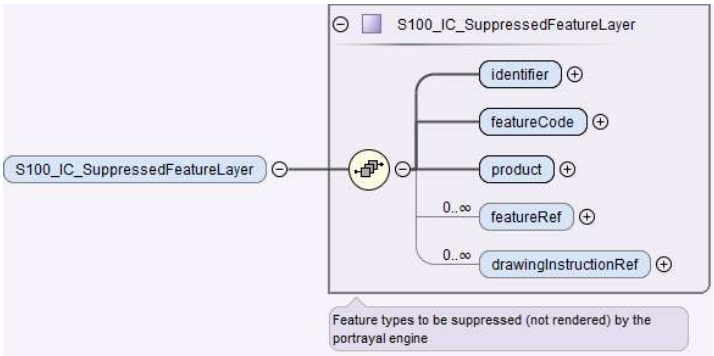


Figure 16-A-8 - Structure of S100_IC_SuppressedFeatureLayer

The **S100_IC_FeatureDerivation** class is abstract and has two derived classes named **S100_IC_SuppressedFeatureInstance** and **S100_IC_HybridFeature.** The elements in each **S100_IC_SuppressedFeatureInstance** and **S100_IC_HybridFeature** correspond to the model described in clause 16-4 and are depicted in Figures 16-A-9 and 16-A-10 below. The contents of these elements are:

- **S100_IC_SuppressedFeatureInstance** element:
  - primary product selectors;
  - secondary product selectors;
  - output product indicators;
  - identifying and descriptive elements (identifier, name, description);
  - reference to optional S100_IC_SimpleRule.
- **S100_IC_HybridFeature** element:
  - primary product selectors;
  - secondary product selectors;
  - output product indicators;
  - identifying and descriptive elements (identifier, name, description);
  - references to one of S100_IC_CompleteRule or S100_IC_ThematicRule.

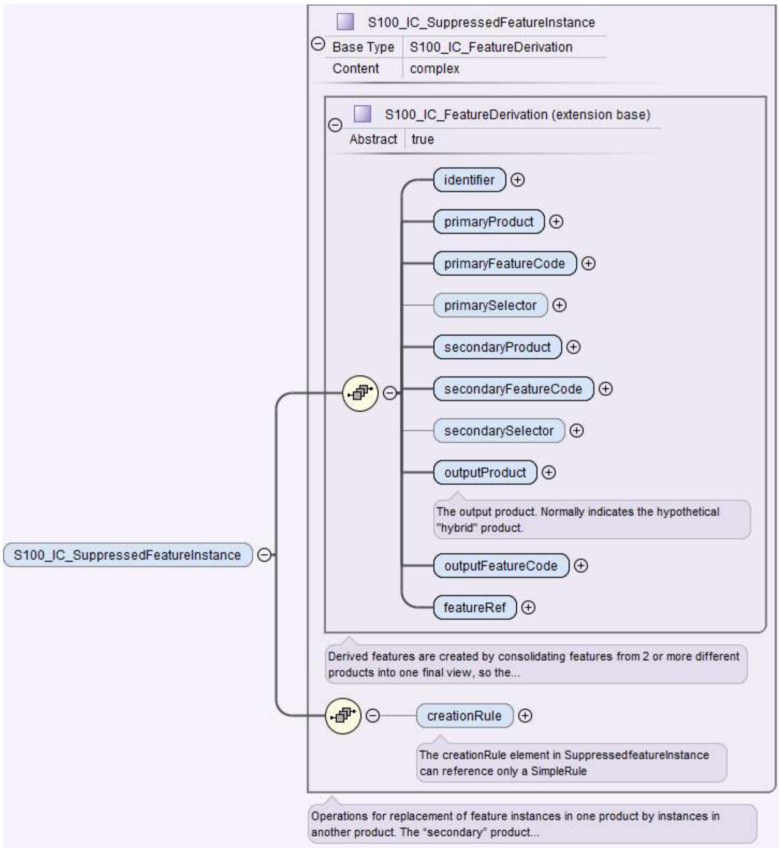


Figure 16-A-9 - Structure of S100_IC_SuppressedFeatureInstance

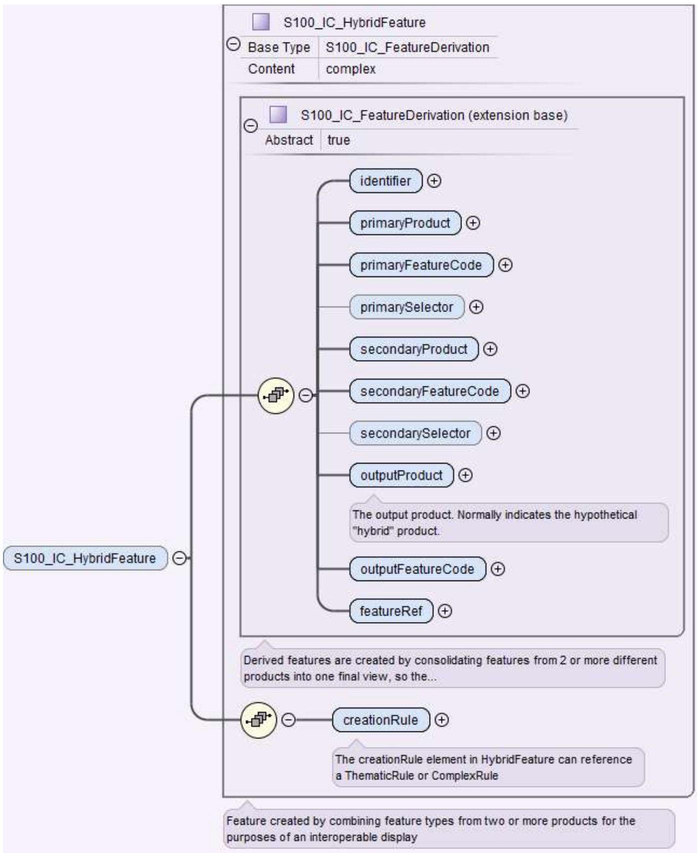


Figure 16-A-10 - Structure of S100_IC_HybridFeature

The **S100_IC_HybridFeatureCreationRule** class is abstract and has three derived classes named **S100_IC_CompleteRule**, **S100_IC_ThematicRule** and **S100_IC_SimpleRule**. The elements in each **S100_IC_SuppressedFeatureInstance** and **S100_IC_HybridFeature** correspond to the model described in clause 16-4 and are depicted in Figures 16-A-11, 16-A-12 and 16-A-13 below. The contents of these elements are:

- **S100_IC_CompleteRule** element:
  - the interoperability level;
  - identifier.

- **S100_IC_ThematicRule** element:
  - the interoperability level;
  - identifier.

- **S100_IC_SimpleRule** element:
  - the interoperability level;
  - identifier.

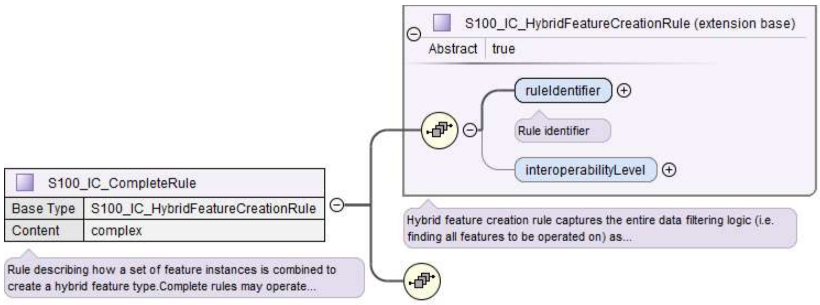


Figure 16-A-11 - Structure of S100_IC_CompleteRule

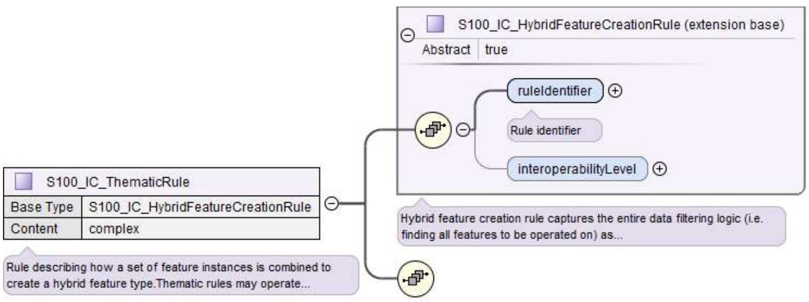


Figure 16-A-12 - Structure of S100_IC_ThematicRule

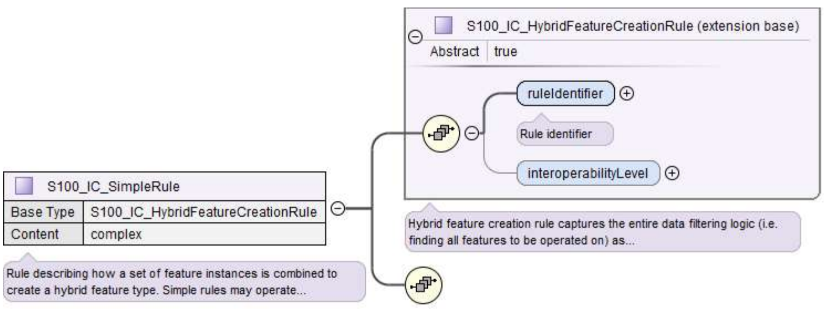


Figure 16-A-13 - Structure of S100_IC_SimpleRule

An example of the XML conforming to the structures is shown in Figure 16-A-14. This figure shows the higher-level structure consisting of the catalogue header elements (*cat:name* through *requirementType*), followed by exemplary display plane and predefined product combinations. The lower-level containers *features*, *drawingInstructions*, and *suppressedFeatureLayers* are shown expanded in Figure 16-A-15 and Figure 16-A-16.

```
<S100IC:S100_IC_InteroperabilityCatalogue xsi:schemaLocation="http://www.iho.int/S100/IC ../../../../../schemas
xmlns:lan="http://standards.iso.org/iso/19115/-3/lan/1.0" xmlns:gco="http://standards.iso.org/iso/19115/-3
xmlns:S100IC="http://www.iho.int/S100/IC">
  + <cat:name>
  + <cat:scope>
  + <cat:versionNumber>
  + <cat:versionDate>
  + <cat:language>
    <description>Example of interoperation catalogue structure</description>
    <comment>This is a hypothetical example showing only the structure of an interoperability catalogue file. T
       actual features in the product(s) indicated.</comment>
    <requirementType>IHO</requirementType>
    <productCovered codeListValue="S-101" codeList="resources/098AA00DICTIONARY.XML"/>
    <productCovered codeListValue="S-102" codeList="resources/098AA00DICTIONARY.XML"/>
  - <displayPlanes>
    - <S100_IC_DisplayPlane>
        <identifier>1</identifier>
        <name>base</name>
        <order>1</order>
        <description>new SOE</description>
      + <features>
      + <drawingInstructions>
      </S100_IC_DisplayPlane>
    + <S100_IC_DisplayPlane>
    + <S100_IC_DisplayPlane>
    + <S100_IC_DisplayPlane>
    </displayPlanes>
  - <predefinedProductCombinations>
    - <S100_IC_PredefinedCombination>
        <identifier>urn:mrn:iho:iop:0.1:pdc:1</identifier>
        <name>ENCandCurrents</name>
        <description>combined S-101 and S-111 - suppressing all ENC current info</description>
        <useConditions>all conditions</useConditions>
        <interoperabilityLevel>2</interoperabilityLevel>
        <includedProduct codeListValue="S-101" codeList="resources/098AA00DICTIONARY.XML"/>
        <includedProduct codeListValue="S-111" codeList="resources/098AA00DICTIONARY.XML"/>
      + <suppressedFeatureLayers>
      </S100_IC_PredefinedCombination>
    </predefinedProductCombinations>
</S100IC:S100_IC_InteroperabilityCatalogue>
```

Figure 16-A-14 - Example of Interoperability Catalogue showing the higher level structure

```
- <features>
  - <S100_IC_Feature>
      <identifier>urn:mrn:iho:s101:1.0:LNDARE</identifier>
      <featureCode>LandArea</featureCode>
      <product codeListValue="S-101" codeList="resources/098AA00DICTIONARY.XML"/>
      <drawingPriority>1</drawingPriority>
      <viewingGroup>20260</viewingGroup>
    </S100_IC_Feature>
  </features>
- <drawingInstructions>
  - <S100_IC_DrawingInstruction>
      <identifier>123456</identifier>
      <featureCode>LandArea</featureCode>
      <product codeListValue="S-101" codeList="resources/098AA00DICTIONARY.XML"/>
      <geometryType>curve</geometryType>
      <drawingPriority>1</drawingPriority>
      <viewingGroup>20260</viewingGroup>
    </S100_IC_DrawingInstruction>
  - <S100_IC_DrawingInstruction>
      <identifier>123457</identifier>
      <featureCode>LandArea</featureCode>
      <product codeListValue="S-101" codeList="resources/098AA00DICTIONARY.XML"/>
      <geometryType>surface</geometryType>
      <drawingPriority>2</drawingPriority>
      <viewingGroup>20260</viewingGroup>
    </S100_IC_DrawingInstruction>
  - <S100_IC_DrawingInstruction>
      <identifier>123458</identifier>
      <featureCode>Current</featureCode>
      <product codeListValue="S-101" codeList="resources/098AA00DICTIONARY.XML"/>
      <geometryType>surface</geometryType>
      <drawingPriority>2</drawingPriority>
      <viewingGroup>20260</viewingGroup>
    </S100_IC_DrawingInstruction>
  </drawingInstructions>
```

Figure 16-A-15 - S100_IC_Feature and S100_IC_DrawingInstruction - XML examples

```xml
<predefinedProductCombinations>
 - <S100_IC_PredefinedCombination>
     <identifier>urn:mrn:iho:iop:0.1:pdc:1</identifier>
     <name>ENCandCurrents</name>
     <description>combined S-101 and S-111 - suppressing all ENC current info</description>
     <useConditions>all conditions</useConditions>
     <interoperabilityLevel>2</interoperabilityLevel>
     <includedProduct codeListValue="S-101" codeList="resources/098AA00DICTIONARY.XML"/>
     <includedProduct codeListValue="S-111" codeList="resources/098AA00DICTIONARY.XML"/>
   - <suppressedFeatureLayers>
     - <S100_IC_SuppressedFeatureLayer>
         <identifier>urn:mrn:iho:iop:0.1:sfl:1</identifier>
         <featureCode>CurrentNonGravitational</featureCode>
         <product codeListValue="S-101" codeList="resources/098AA00DICTIONARY.XML"/>
         <featureRef>urn:mrn:iho:s111:1.0:CURENT</featureRef>
       </S100_IC_SuppressedFeatureLayer>
     </suppressedFeatureLayers>
   </S100_IC_PredefinedCombination>
```

Figure 16-A-16 - S100_IC_PredefinedCombination and S100_IC_SuppressedFeatureLayer - XML examples

Figure 16-A-17 shows an expanded example of a display plane showing one of the display planes for S-101 – this one contains navigation aids. The specification for one feature has been expanded as an example.

The drawing priority and viewing group given in a Portrayal Catalogue can be overridden for interoperation purposes by different values encoded in an Interoperability Catalogue, on a per-feature (type) basis.

```xml
<S100_IC_DisplayPlane>
   <identifier>2</identifier>
   <name>baseBathy</name>
   <order>2</order>
   <description>all S-101 bathy features are located here</description>
 - <features>
   - <S100_IC_Feature>
       <identifier>urn:mrn:iho:s101:1.0:DEPARE</identifier>
       <featureCode>DepthArea</featureCode>
       <product codeListValue="S-101" codeList="resources/098AA00DICTIONARY.XML"/>
       <drawingPriority>1</drawingPriority>
       <viewingGroup>20280</viewingGroup>
     </S100_IC_Feature>
   - <S100_IC_Feature>
       <identifier>urn:mrn:iho:s101:1.0:DEPCNT</identifier>
       <featureCode>DepthContour</featureCode>
       <product codeListValue="S-101" codeList="resources/098AA00DICTIONARY.XML"/>
       <drawingPriority>2</drawingPriority>
       <viewingGroup>20280</viewingGroup>
     </S100_IC_Feature>
   </features>
```

Figure 16-A-17 - Example of display plane with S-101 features (informative)

Figure 16-A-18 depicts an example of the use of the substitution capabilities of an Interoperability Catalogue. The element substituteSymbolization for S-101 feature CurrentNonGravitational has its point and line symbols replaced by new symbols which are identified inline. The symbol and linestyle files are included in the Portrayal Catalogue identified by IHOICPCEXMP0001.

```xml
<S100_IC_DrawingInstruction>
    <identifier>4A</identifier>
    <featureCode>CurrentNonGravitational</featureCode>
    <product codeListValue="S-101" codeList="resources/098AA00DICTIONARY.XML"/>
    <drawingPriority>1</drawingPriority>
    <viewingGroup>20240</viewingGroup>
  - <substituteSymbolization>
    - <pc catalogid="IHOICPCEXMP0001">
      - <symbol id="CURENTZ1">
        - <description>
            <name>CURENTZ1</name>
            <description>non-tidal current (replacement symbol)</description>
            <language>en</language>
          </description>
          <fileName>CURENTZ1.svg</fileName>
          <fileType>Symbol</fileType>
          <fileFormat>SVG</fileFormat>
        </symbol>
      - <lineStyle id="CURENTZ1">
        - <description>
            <name>CURENTZ1</name>
            <description>non-tidal current</description>
            <language>en</language>
          </description>
          <fileName>CURENTZ1.xml</fileName>
          <fileType>LineStyle</fileType>
          <fileFormat>XML</fileFormat>
        </lineStyle>
      </pc>
    </substituteSymbolization>
```

Figure 16-A-18 - Example of substitution in a drawing instruction (informative)

16-A-3 Location of Schema files

The schema files are available from the locations listed in Table 16-A-1 below.

Table 16-A-1 - Schema files locations

File	Description	Version	Location
S100_IC.xsd	XML Schema for Interoperability Catalogue	5.0	https://github.com/IHO-S100WG
S100_IC.sch	Schematron file for validating Interoperability Catalogue	5.0	https://github.com/IHO-S100WG

NOTE: The XML Schema as distributed imports ISO metadata Schemas from the ISO Internet location encoded in the S100_IC.xsd file. Implementers may wish to store the ISO metadata Schemas locally and use the local installation instead.

Page intentionally left blank

Appendix 16-B

Guidance on How to Make Product Specifications Interoperable

(informative)

16-B-1 How to make Product Specifications interoperable

This appendix gives guidelines for how to identify concepts that should be factored into the development of an Interoperability Catalogue and suggestions for how to write rules to address associated interoperability issues. Moreover, these guidelines can be useful in the development of Product Specifications that will be included in an Interoperability Catalogue and its specification, in order to make these ready for interoperability in user systems.

Due to the interconnected nature of the Product Specifications that are under an interoperability Schema, a form of overarching change and content management is a necessity. It is recommended that a management structure aimed at harmonization of Product Specifications that are under an interoperability Schema is established.

For portrayal considerations, see Clause 16-11.

For interoperation requirements for Product Specifications, see Clause 16-C-3.

16-B-2 Duplicated features

Perhaps the most significant issue to deal with when considering interoperability is how to deconflict duplicated features between layers. The following paragraphs deal with major categories of duplicate features.

16-B-2.1 Duplicated features, same model

Where there are equivalent models with same feature concept and attribute bindings, there still may be different attribute values due to issues with maintaining the same update sequence between different products. Both the production and end user sides can impact this issue. When considering this issue in the creation of an Interoperability Catalogue, priority should be given to the product that is most likely to be up to date with the latest information.

Developers of Product Specifications that are expected to be used in a system in interoperability mode should consider if the features within the specifications are likely to be more frequently updated than those products that may serve as a base layer or base layer combination for the product being developed. These considerations should be factored in when describing the production of the product and envisioned future use of the product.

IC_SuppressedFeatureLayer elements only have feature code and product as attributes for suppression, this means that all instances of a listed feature class will be suppressed. This is important to remember when creating rules that promote alternative instances. IC_Feature and IC_DrawingInstruction can have attribute combinations and spatial primitives to select the alternative instances. There is therefore, a risk that unless sufficient attention to details is given, important instances may be omitted.

EXAMPLE: If **Restricted Area Navigational** in one product is suppressed, and **Restricted Area Navigational** with attribute **category of restricted area = 4 (nature reserve)** in another product is promoted in its place, there is a risk that only instances with that combination will be visible, and all others supressed.

16-B-2.2 Duplicated features, different models

Where the feature concept, attribute bindings, and values of selected attributes are mostly the same but there are minor differences in the different products, such as extra attribute bindings, developers of an Interoperability Catalogue should consider which version of the feature is the higher value for the end user, and give that version priority. There may be different answers depending on the operational situation that a predefined combination tries to support, and this must be considered as part of constructing an Interoperability Catalogue.

EXAMPLE: Interoperability Catalogue developers compare the specifications and data samples of **Pilot Boarding Place** features from a “Piloting Information” data product and the Pilot Boarding Place features in another product and decide that features from the “Piloting Information” datasets have more value in approach and harbour entry scenarios.

Developers of Product Specifications should consider how their data model is similar and/or different from other related Product Specifications and the justification for this; and make recommendations to the developers of an Interoperability Catalogue for how to best select between the versions of related features.

16-B-2.3 Duplicate feature domains

Where feature concepts are different, but the information content is equivalent, considerations should include the update cycle of the information and when creating an Interoperability Catalogue, priority should be given to the concept that is most likely to be updated most frequently. Other considerations should include any relations that the concepts has with other feature concepts, and any consequences of breaking these must be considered when choosing which concept to give the priority and which concept to suppress.

EXAMPLE: Developers investigate the update cycles of real-time current data products and discover that they are updated more frequently than a chart product with **Current – Non-Gravitational** and **Tidal stream – Flood/Ebb** features, and features from the real-time current datasets are therefore preferred over the chart product current features. Note that the question is decided not by comparing dates encoded in features, but on the basis of real-time data that is available on an ongoing basis vs. historical information gathered at a past date.

Where it is not practical to avoid concept overlaps, Product Specification developers should strive to maintain a data model that is as harmonized with related data models as possible. Due considerations should be taken before developing a concept that is different but functionally equal to similar concepts in other Product Specifications.

16-B-2.4 Geometry

The geometry of a feature is a significant attribute that must be considered when developing an Interoperability Catalogue, as it defines the ‘where-part’ of the feature object. Similar to other attributes, decisions may have to be made to address issues such as select one feature over another, for example where types in one product is affected by small scale, while another by large scale, or if merging the two is a better approach. The following paragraphs give more details about options for deconflicting geometry between products with the help of an Interoperability Catalogue.

16-B-2.4.1 Combined geometry

Where there is a feature in one dataset that effectively augments the geometry of a conceptually different feature in another dataset, developers of an Interoperability Catalogue need to specify a hybrid feature with portrayal that can correctly portray the combined information. Moreover, it should be considered if it is appropriate for clarity in the user system to supress the features in the origin datasets.

EXAMPLE: A dredged area augmented with high definition bathymetry from survey of recent dredging operations giving more water and wider area than dredged area in a navigation product, combine to give a bigger (new boundary) dredged area than present in the navigation product.

Developers of Product Specifications that may result in hybrid features when interacting with certain other products, should cooperate with the developers of the related Product Specifications to correctly define the conditions for appropriate use of hybrid features, and communicate these specifications to the Interoperability Catalogue developers.

16-B-2.4.2 Spatial discrepancy, unrelated to scaled or cartographic smoothing

If the same feature instances exist in different data products, and are expected to have discrepant geometries, developers of an Interoperability Catalogue should establish the cause of the discrepancy. The cause will likely influence the solution to be implemented in the specific Interoperability Catalogue. For example, if one of the data products has more detailed information due to differences in the scopes of the Product Specifications, the developers of an Interoperability Catalogue should consider suppressing the feature class in the less-detailed product giving preference to the same feature class in the product with greater detail. On the other hand, if the discrepancy is found to occur irregularly, it may be more appropriate to suppress the less-detailed instances and prefer the more-detailed instances, independently of the data products to which they belong.

Example: A navigation product has Restricted Area features, while an environmental product has Marine Protected Area features which show greater details for a sub class of restricted areas.

Developers of Product Specifications should examine their Specification scope and consider if it is likely that resulting data products include information that will be superior or inferior to the same, or similar, information in other products. For example, if information is only for contextual purposes, it is likely that better information is available in another product and in an interoperability ready system these contextual features should be supressed in the presence of more accurate information. Such expectations should be communicated with the Interoperability Catalogue developers where the products are included.

16-B-2.4.3 Spatial discrepancies, related to scale or cartographic smoothing

Where there is the expectation of differences of geometry for same feature instance in different products it is important to establish the cause, as this will likely impact the solution implemented in an Interoperability Catalogue. If the cause is related to scale or cartographic smoothing in one product over another due to issues such as different scopes (intended use) of the Specifications, the Interoperability Catalogue developers should consider supressing the lesser detailed product and promoting the product with greater detail. If it is irregular occurrences of spatial discrepancies, it may be appropriate to supress the instances where there are less detail and promote the more detailed instances.

EXAMPLE: A navigation product has approximate surface current instances using climatic data that is omitted at smaller scales; A surface current gridded data product has greater details and is daily updated. The surface current information in the navigation product may be suppressed to give preference to the information from the surface current gridded data product

Developers of Product Specifications should examine the Specification scope and consider if it is likely that resulting data products include information that will be superior or inferior than the same information in other products. For example, if information is only for contextual purposes or is reduced in detail due to scale, it may be that better information is available in another product and in an interoperability ready system these lesser detailed features should be supressed in the presence of more accurate information. Such expectations should be communicated with the Interoperability Catalogue developers where the products are included.

16-B-2.5 Display of text

For details about display of text, including placement, display selection and management of long text, see Clause 16-11.5. The Interoperability Catalogue does not address these issues in general terms.

Developers of Product Specifications should be aware that the instructions they place within the Product Specification generally carry through even when the product is used in a user system in interoperability mode. Moreover, text placement issues in interoperability mode are expected to occur at the border between two products which may result in text being partially obscured due to priority issues. Developers of user systems with interoperability capabilities will likely have experience with solving such issues and advice should be sought with them in how to mitigate such issues. Additionally, Part 16-11.5 recommends OEMs to provide functionality that seek to address most of the issues that cause text to be partially obscured.

16-B-2.6 Skin of the Earth replacement

Some systems implement a Skin of the Earth concept that require the system to always provide continuous coverage of select data elements to make up a surface of the earth. These data elements are usually drawn first in portrayal processing. In some circumstances, it may be necessary to replace some of these select data elements with more appropriate data. In the interoperability model, Skin of the Earth replacement is a specialization of combined geometry, see Clause 16-B-2.4.1 for details.

16-B-2.6.1 Skin of the Earth feature adjusting

When combining feature instances into new Skin of the Earth feature instances, considerations should be given to the attributes of the resulting Skin of the Earth feature, as a combined feature may have altered geographical representation, attribute combinations or attribute values that may impact its inclusion in the Skin of the Earth.

EXAMPLE: Shoaling in a channel in a navigation product may be indicated by a high definition bathymetry product, and a shallower channel hybrid feature replaces the navigational product feature instance, which also has an amended shape. Adjacent depth areas grow due to the shoaling.

16-B-2.7 Blended feature concepts

Blended feature concepts or blended portrayals can be produced by using transparency between related features; or creating a temporary blended feature; or blended portrayal (rule and/or symbol) of specific combinations of features from different products. See 16-11.6 for portrayal considerations and example of use case. Blended concepts will typically be created by using **S100_IC_PredefinedCombination** which link to a hybrid Portrayal Catalogue that includes the features to be combined and a suppression rule, for example by using **S-100_IC_SuppressedFeatureLayer**, for the features that are to be replaced.

Developers of Product Specifications that are likely to be used in blended feature concepts by ECDIS in interoperability mode should communicate their intentions with developers of related Specifications so that awareness is created about the inter-dependencies of these types of relationships. Such communication is especially important when revisions to these Specifications are considered. Doing so will help manage risks to breaking the relationships as the related Product Specifications transition through their life cycle.

16-B-2.8 Hierarchy of data

In this context, hierarchy of data means the stacking of data products (layers) within a predefined combination. Predefined combinations are generally created with a particular type of operational view in mind, and therefore the hierarchy of data may vary between predefined combinations.

For example, in ECDIS, the ENC will be the base layer; that is, the lowest layer in a predefined combination.

16-B-2.8.1 Predefined combinations

Predefined combinations are used to define the hierarchy of data between different S-100 based specifications. An instance of **IC_PredefinedCombination** is associated to **IC_DisplayPlane** instances to give the hierarchy of the data products that are intended to be used. The attribute **displayPriority** within the **IC_DisplayPlane** gives the order in which the layers are drawn. See 16-4.4.2.6 for more details on predefined combinations.

16-B-2.9 New datasets

New datasets that are added to a system with interoperability mode will be managed in accordance to any loaded Interoperability Catalogue that includes the Product Specification that relevant data product is conformant with. Data producers should therefore be encouraged to perform sufficient tests to ensure new datasets perform as envisioned in an interoperability environment.

16-B-2.9.1 New datasets – coverages

New datasets may alter the available coverages of particular data that is used for interoperability views; and any new dataset should therefore be sufficiently tested to ensure performance is as envisioned.

16-B-2.9.2 New datasets – maximum and minimum display scales

New datasets may alter the available data in particular scales and/or scale bands, for example, by adding or removing data coverage. Considerations should therefore be given to harmonization of maximum and minimum display scales when a new dataset is produced.

16-B-2.9.3 New datasets – feature geometry

New datasets may alter the available feature geometry of available data that is used for interoperability views. Changes include extending or reducing size of areas, changing geometry type from area to point, point to area, area to line or line to area. Therefore, any new dataset should be sufficiently tested to ensure performance is as envisioned.

16-B-2.9.4 New datasets – types and attributes

New datasets may change type and attributes of instances in the user system, for example a platform may be removed and an obstruction remain. These changes may impact the situational view created by the Interoperability Catalogue as changes to feature classes and attribute combinations may mean objects are no longer covered by conditions specific to a predefined combination, or new objects are new covered. Therefore, any new dataset should be sufficiently tested to ensure performance is as envisioned.

16-B-2.10 Dataset loading and unloading due to scale

Developers of Product Specifications and producers of data should make every effort to harmonize effects of maximum and minimum display scales at loading/unloading time between related product to control over-scale indicators and datasets, in order to avoid situations where one overlay is in scale but not another.

16-B-2.11 New data products

When a new product is added to an existing Interoperability Catalogue, a new version will be required, see 16-10.4. During the development of the new version, the Interoperability Catalogue developers will need to review existing predefined combinations for impact in addition to developing the new predefined combinations to manage the situational views that the new product is intended for. Close coordination between the authors of the Product Specification and the Interoperability Catalogue producer is highly recommended to ensure all relevant changes to the Interoperability Catalogue are accounted for.

16-B-2.12 Metadata

This edition of the Specification does not provide for comparing information that is not encoded as attributes of feature (or information type) instances. This means that metadata cannot be used in interleaving, filters, or rules unless it is encoded in feature attributes (for example “horizontal position uncertainty” attributes) or meta-features (for example **Quality of Bathymetric Data**). If Product Specification authors envisage a need to use metadata in interoperability, the Product Specification Application Schema should be designed so as to make the relevant metadata available as feature attributes or meta-features.

Note that using information from meta-features in interoperability operations may involve spatial operations.

16-B-3 Portrayal distinguishability – colour set-asides

Special consideration must be made when creating portrayal rules related to colour choices for a Product Specification. Depending on the context certain colours have specific meaning. For example, for marine navigation, IHO S-4 gives indications for magenta line meaning something non-physical, while black colour implies a physical item. See IHO S-4, clauses B-141 to B-145 for additional details.

Example: on ENC the light sectors marking intricate inshore channels in, such as in Scandinavian waters are shown in red, green, and yellow.

16-B-3.1 Black (S-4 – B-141)

Black is normally used for all physical (solid) features, including depth information.

16-B-3.2 Magenta (S-4 – B-142)

The general principles for the use of magenta are that it should be reserved for:

- Drawing attention to symbols for features which have a significance extending beyond their immediate location.
- Distinguishing information superimposed on the physical features and not implying any permanent physical obstruction (see S-4 clause B-145 for the use of green for environmental information).

16-B-3.3 Buff (yellow) or grey (S-4 – B-143)

A colour, usually buff (yellow) or grey, must be used as a land tint in paper charts. ENC portrayed with S-52 in an ECDIS uses a yellow/brown colour (LANDA).

16-B-3.4 Blue (S-4 – B-144)

The colour blue has been used as a tint to emphasize shallow water. Two (or more) densities of blue tint may be used to show different depth bands of shallow water, the darkest tint showing the shallowest water.

16-B-3.5 Green (S-4 – B-145)

The colour green may be used as a tint for inter-tidal areas. Green may also be used, instead of magenta, for environmental information and limits; see S-4 clause B-437.2b.

16-B-3.6 Red

It should be noted that any symbology using red may be an issue for navigation equipment operated in night mode, and therefore the use of red should be avoided as much as possible.

16-B-3.7 Day/night/dusk modes

It may be required that for every Product Specification that is intended for a navigation system to have colours specified for day, dusk, and night modes. The system may be required to utilize these colours depending on the mode the viewing system is set to.

16-B-4 Rendering order steps

Product Specifications that are in scope of an Interoperability Catalogue should have at least ten rendering order steps between display groups. This allows more flexibility when configuring interleaving with other products in an Interoperability Catalogue.

Appendix 16-C
User Interaction Constraints and Expectations
(informative)

16-C-1 Introduction

S-100 based products are developed for a variety of uses but generally define only the product itself and its use, not how it will be used in in combination with other S-100-based data products or sensor information. It is anticipated that most S-100-based systems, like ECDIS and other ship and shore-based systems, will use several different products simultaneously, each providing one or more information layers. Other data layers such as real-time sensor information from sources like radar and AIS are also expected to be present. The smooth interoperation and harmonized user-friendly graphical presentations of these various products is necessary for safety and efficiency. This annex gives some guidance to principles that should be considered when developing smooth interoperation and harmonized graphical presentation of S-100 data products

16-C-2 Structured interoperation

IMO has issued recommendations and guidelines on how to present navigation related information. Of particular significance are the documents noted below.

MSC.191(79), Recommendation on Performance Standards for the Presentation of Navigation-Related Information on Shipborne Navigational Displays, specify the presentation of navigational information on the bridge of a ship, including the consistent use of navigational terms, abbreviations, colours and symbols, as well as other presentation characteristics. It also addresses the presentation of navigation information related to specific navigational tasks by recognizing the use of user selected presentations in addition to presentations required by the individual performance standards adopted by the Organization.

MSC.1/Circ.1512, Guideline on Software Quality Assurance and Human-Centred Design for E-Navigation, gives guidance in achieving trustworthy software and usability in the development of complex systems requires a disciplined and structured approach. The guideline encourages particular focus on Software Quality Assurance (SQA) and Human-Centred Design (HCD) that includes Usability Testing (UT). Systems so designed, developed and managed throughout their life cycle deliver improved user performance, being stable and resilient, and, most importantly, support users in low and high workload environments, such as during challenging navigation and environmental conditions when users are most vulnerable to making mistakes and when error management and recovery is essential. Other important benefits include limiting the amount of operator familiarization training that is needed and the time and resources required for system maintenance and support.

MSC.1/Circ.1609, Guidelines for the Standardization of User Interface Design for Navigation Equipment, apply to Integrated Navigation Systems (INS), Electronic Chart Display and Information Systems (ECDIS) and radar equipment. The guidelines may also be useful to other electronic navigation equipment, and navigation sensors to improve standardization and usability. The aim of the Guidelines is to promote standardization of user interfaces to help meet user needs.

IMO SN.1/Circ.243/Rev.2, Guidelines for the Presentation of Navigational-Related Symbols, Terms and Abbreviations, stems from a compelling user need for greater standardization to enhance usability across navigation equipment and systems. Significant variation between systems and equipment produced by different manufacturers has led to inconsistency in the way essential information is presented, understood and used to perform key navigation safety functions. Improved standardization of navigation systems will provide users with more timely access to essential information and functions that support safe navigation.

16-C-3 Interoperation requirements for Product Specifications

Responsible parties for Product Specifications that are included in an Interoperability Catalogue should consider the impact on this Interoperability Catalogue and associated Product Specifications throughout the lifecycle of the Product Specification. The general principles of Software Quality Assurance (SQA) as found in section 4 of the Annex to MSC.1/Circ.1512 should be applied.

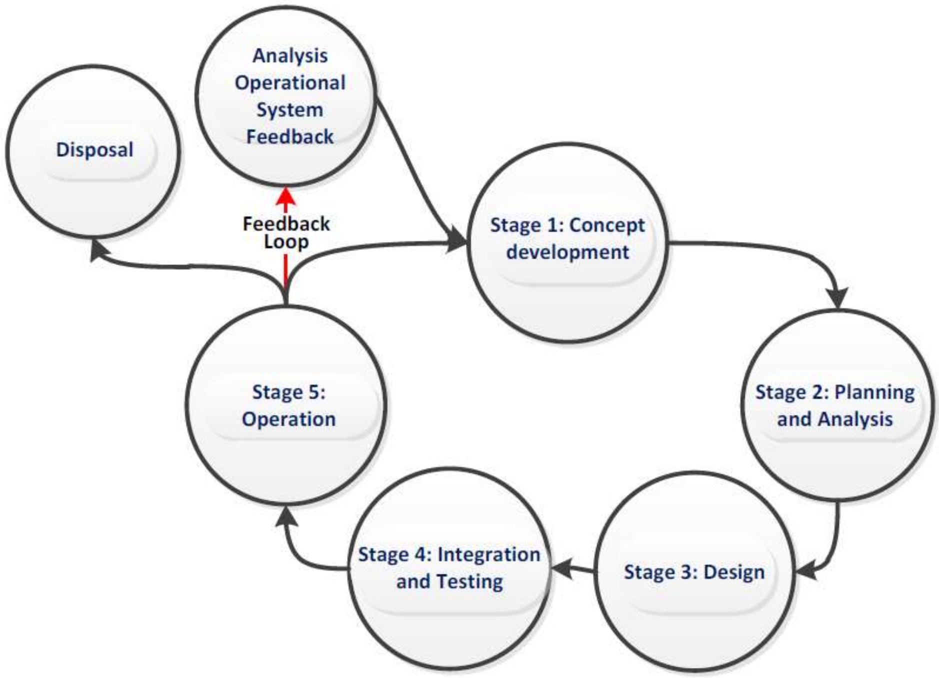


Figure 16-C-1- Generic life cycle (from MSC.1/Circ.1512)

Figure 16-C-1 shows a typical generic life cycle with the stages recommended as a minimum for the management of the development of Product Specifications that are used with the Interoperability Catalogue:

- Analysis of operational system feedback;
- Stage 1: Concept development;
- Stage 2: Planning and analysis;
- Stage 3: Design;
- Stage 4: Integration and testing;
- Stage 5: Operation; and
- Disposal.

16-C-4 Customization – user level

S-100 compatible systems that support the IHO Interoperability Catalogue may include functionality that allow end users (including ship owners, operators, and shipboard officers) to add new predefined combinations according with their needs. These added combinations must not interfere or degrade the official IHO Interoperability Catalogue functions.

16-C-5 Support Human-Centred Design

As noted in MSC.1/Circ.1512, HCD helps to ensure that human factors-related knowledge and techniques in system design and development processes are addressed, thus ensuring that user needs and safety are met. Implementers of this Specification should perform Usability Testing (UT) and follow the principles stated in MSC.1/Circ.1512 when designing the user interface for interoperability in ECDIS, including the following HCD activities that are carried out to inform development throughout the life cycle:

- Pre-activity: Conduct Early Human Element Analysis (EHEA);
- Activity 1: Understand and specify the context of use;
- Activity 2: Identify the user requirements;
- Activity 3: Produce and/or develop design solutions to meet user requirements;
- Activity 4: Evaluate the design against usability criteria; and
- Activity 5: Maintain operational usability.

Note that fundamental to HCD is the collection of user feedback through Usability Testing.

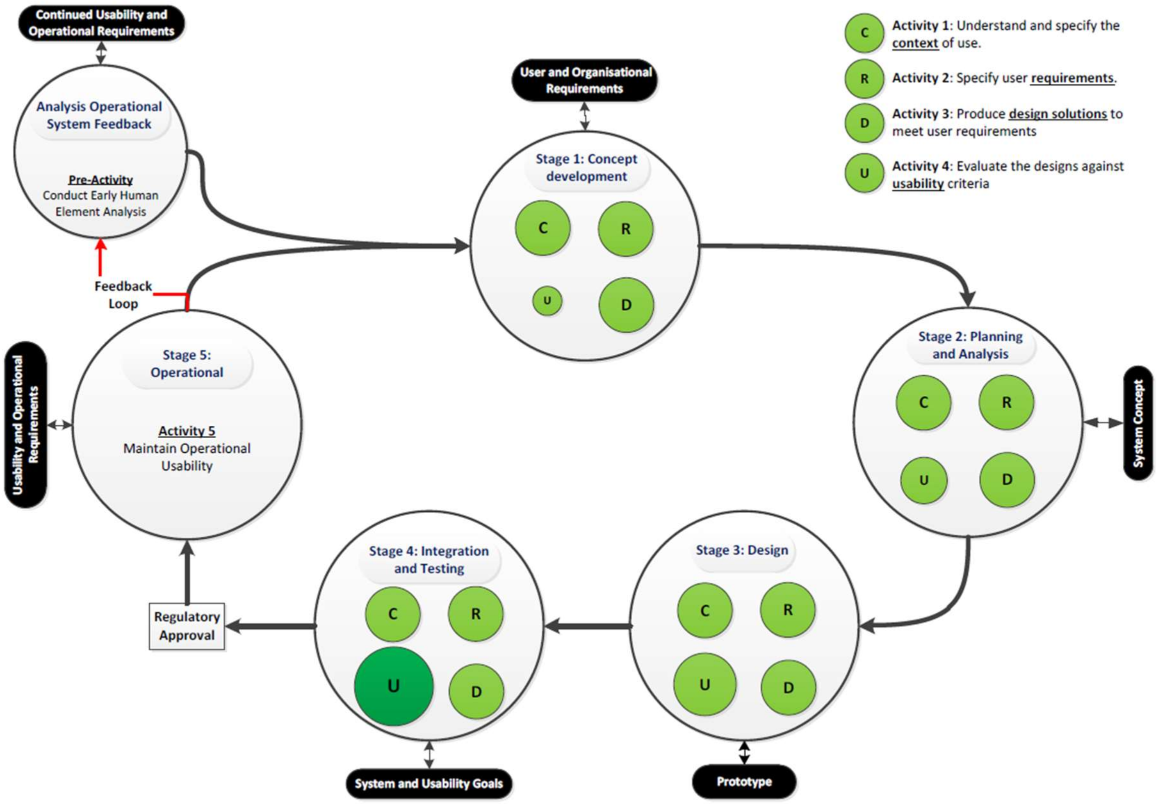


Figure 16-C-2 - Overview of HCD for e-navigation systems (from MSC.1/Circ.1512)

The details of recommended activities at each stage are found in section 6 of the Annex to MSC.1/Circ.1512.

16-C-5.1 Human-Centred Design and user interface

MSC.1/Circ.1609, Guidelines for the Standardization of User Interface Design for Navigation Equipment notes in section 15 principles utilized in the formation of the guidelines. These and the resulting appendixes may be of benefit for any system implementing support for an Interoperability Catalogue.

16-C-5.2 Human-Centred Design and text

MSC.191(79) as amended, notes in section 5.2.3 that use of text should be presented using simple unambiguous language that is easy to understand. Navigation terms and abbreviations should be presented using the nomenclature defined in IMO SN.1/Circ.243/Rev.2 and MSC.1/Circ.1609 and encouraged their use for all shipborne navigational systems and equipment. Shore based equipment that interact with shipborne system may also benefit from guidance in these guidelines.

16-C-6 User control over loaded set

Users of systems with interoperability capabilities may have functionality to load additional products, or turn off one or more of the data products in a predefined combination.

Data from such additional products which are not mentioned in an Interoperability Catalogue should be treated by the system according to the priorities and viewing groups encoded in the product's own Portrayal Catalogue (for example interleaved with layers from products controlled by the Interoperability Catalogue according to their relative rendering orders and drawing priorities).

When a user turns off a data product, the portrayal should treat it as if the relevant datasets are not available on the system at all. For example, interoperability rules that are made inapplicable due to one of the data products in their conditions being turned off are unavailable.

16-C-7 User control over interoperation level

The system should allow the user to change the interoperation level (see 16-8) and/or pick a predefined combination by means of simple operations. Any options offered to the user must be valid in context; for example, if the Interoperability Catalogue supports more than one interoperability level, the user interface should offer the user a choice of predefined combinations at an interoperability level. The listed combinations should be only those defined at that level in the Interoperability Catalogue.

The system should minimise demands for user interaction when changing interoperability level or predefined combinations, subject to constraints imposed by the platform and interface. Some implications of this guidance are:

1) When the interoperability level alone is changed and the Interoperability Catalogue contains a predefined combination of the new level that lists the currently displayed product set, the system should apply the rules of the new level to the product set immediately. Alternative predefined combinations for the level may be offered in an unobtrusive way.

   EXAMPLE: Product A and Product B data are both on-screen when Level 1 (no suppression) is changed to Level 2 (suppression supported), and the catalogue includes a "Level 2 Product B + Product A" predefined combination. Interleaving of Product A and Product B features (Level 1) immediately changes to suppression of Product A navigation aids by Product B navigation aid features (Level 2).

   Optionally, an indication may be provided to inform the user of the active predefined combinations defined at the new level in the Interoperability Catalogue.

2) When the predefined combination alone is changed and the Interoperability Catalogue contains the new predefined combination at the current level, the system should apply the rules of the current level to the new predefined combination.

   EXAMPLE: The system is in Level 2 (suppression supported) and the Product B + Product A predefined combination is changed to Product B + Product A + Product C. The system suppresses Product A Restricted Area features of type "nature reserve" in favour of Marine Protected Areas from Product C.

3) When the Interoperability Catalogue does not contain a predefined combination at a newly selected level, the user interface should provide an indication of this to the user (though not necessarily by disabling the choice or blocking the transition). Strategies for dealing with this situation are left to interface designers. For example, systems may offer to use the closest fit in an Interoperability Catalogue with any residual on-screen products as ordinary overlays.

16-C-8 Priority overrides for user-specified settings

Where user action amends a setting, which then conflicts with a system setting, the user setting should override the system setting. The system may give indication of this override.

EXAMPLE: Feature rendering order set by a user should override settings in an Interoperability Catalogue or Portrayal Catalogue.

Appendix 16-D

Data Encoding Guide

(Normative)

16-D-1 Introduction

This appendix contains encoding guidance on syntax, content, and catalogue structure for Interoperability Catalogue developers. Guidance on how to make Product Specifications interoperable and what principles to apply when developing an Interoperability Catalogue is provided in 16-B. For definitions of catalogue elements, their attributes, and associations, refer to 16-4.

16-D-2 General encoding notes

16-D-2.1 Identifiers and references

Several catalogue elements have an *identifier* attribute. The value of this attribute must conform to the syntax for a Uniform Resource Identifier (URI). This means it may be a URL, an integer, alphanumeric character string without whitespace, or a URN. Any additional restrictions are mentioned in the encoding notes for the appropriate element, which make up the rest of this appendix.

Some catalogue classes in the catalogue data model (16-4) have associations that act as references to the element at the other end of the association. This is encoded in the data format as an XML child element of the referrer, whose XML tag is the same as the role name in the UML model. The value of such a reference must be equal to the value of the *identifier* attribute of the referenced element.

EXAMPLE:

The tag `<featureRef>urn:mrn:iho:chart:CURENT</featureRef>` in an **S100_IC_SuppressedFeatureLayer** element is a reference to the **S100_IC_Feature** element with tag

`<identifier>urn:mrn:iho:chart:CURENT</identifier>`.

16-D-2.2 Feature codes

Some catalogue elements have a *featureCode* attribute. The value of attribute *featureCode* must be the camel case code of the feature as encoded in the Feature Catalogue for the product named in the *product* co-attribute.

16-D-2.3 Element S100_IC_InteroperabilityCatalogue

Any product mentioned in any attribute of type *dataProduct* of a catalogue element (for example **S100_IC_Feature.***product*, **S100_IC_PredefinedCombination.***includedProduct*, etc.) must also be mentioned in a *productCovered* attribute of this element.

The attribute *productCovered* will be used by the system in deciding whether to apply interoperability rules or fall back on ordinary overlay portrayal and therefore all products taken into account when developing the catalogue must be listed.

The *name*, *description,* and *requirementDescription* attributes should be populated with text values of appropriate size that are meaningful to people developing, configuring, or using Interoperability Catalogues, including end users. These attributes, especially *requirementDescription,* will potentially be displayed to end users when they select an Interoperability Catalogue for a particular task supported by a system, and should be populated with this use in mind.

The attribute *interoperabilityLevel* may be used to indicate highest level of interoperability functionality encoded within the catalogue.

16-D-2.4 Element S100_IC_DisplayPlane

A **S100_IC_DisplayPlane** element must contain at least one instance of **S100_IC_DrawingInstruction** or **S100_IC_Feature**. It may contain multiple instances of either or both. The choice depends on whether symbols or other components of drawing instructions are being substituted.

The *name* and *description* attributes should be populated with text values that indicate the purpose and feature content of the display plane to people developing, configuring, or using Interoperability Catalogues, including end users.

16-D-2.5 Element S100_IC_DrawingInstruction

Drawing instruction elements in the Feature Catalogue override the drawing instructions generated directly from the data product's Portrayal Catalogue. Details of this overriding are described in clause 16-4.4.2.4.

The attributes *product*, *featureCode*, *geometryType*, and *attributeCombination* values together make up a filter condition determining the subset of instances of a feature type to which the drawing priority and viewing group encoded in **S100_IC_DrawingInstruction** apply. They are to be applied in conjunction ("AND") – that is, the **S100_IC_DrawingInstruction's** viewing group and drawing priority apply only when the conditions expressed by all these attributes are satisfied. (Attributes *geometryType* and *attributeCombination* being optional are ignored if not encoded.)

If an instance of **S100_IC_DisplayPlane** contains both **S100_IC_Feature** and **S100_IC_DrawingInstruction** elements with the same "filter condition", their drawing priority and viewing group must also be the same.

Features (drawing instructions) not satisfying a filter condition in an instance of **S100_IC_DrawingInstruction** are treated according to any other interoperability rules which may apply to them, or if none, they treated according to the data product's Portrayal Catalogue.

Distinction: S100_IC_Feature.

16-D-2.6 Element S100_IC_Feature

S100_IC_Feature elements in the Interoperability Catalogue override the drawing priority and viewing group in the data product's Portrayal Catalogue. Details of this overriding are described in 16-4.4.2.3.

S100_IC_Feature elements have the same four filter condition attributes as **S100_IC_DrawingInstruction** and the same rules and constraints described in 16-D-2.4 apply.

Distinction: S100_IC_DrawingInstruction.

16-D-2.7 Element S100_IC_DisplayPlane

As noted in 16-D-2.5 and 16-D-2.6 an instance of **S100_IC_DisplayPlane** must contain at least one of **S100_IC_Feature** or **S100_IC_DrawingInstruction**.

16-D-2.8 Element S100_IC_PredefinedCombination

A **S100_IC_PredefinedCombination** includes at least two different data products. The attribute *includedProduct* must be populated with all data products referenced directly or indirectly in this predefined combination, including:

- The *product* attribute of a **S100_SuppressedFeatureLayer** contained in this element;
- The *product* attribute of a **S100_Feature** or **S100_IC_DrawingInstruction** referenced by an **S100_IC_SuppressedFeatureLayer** element contained in this element.
- When a **S100_SuppressedFeatureLayer** is included, a reference to the replacement **S100_IC_DrawingInstruction** should also be present.

The *name*, *description,* and *useConditions* attributes should be populated with text values of appropriate size that are meaningful to people developing, configuring, or using Interoperability Catalogues, including end users. These attributes, especially *useConditions,* will potentially be displayed to end users when they select an Interoperability Catalogue for a particular task supported by a system, and should be populated with this use in mind.

16-D-2.9 Element S100_IC_SuppressedFeatureLayer

The *featureCode* and *product* attributes identify a feature type which will be suppressed in its entirety; that is, all instances of the feature from that product will be hidden. They will be replaced by instances of the feature type and product indicated by the referenced **S100_IC_Feature** (or **S100_IC_DrawingInstruction**). Both elements may include conditions pertaining to attribute values and geometry type, as described in clause 16.5. The implications should be carefully considered when referencing instances of **S100_IC_Feature** or **S100_IC_DrawingInstruction**, for example:

- Consider whether all feature instances of the indicated feature type from the replacing product will be displayed, or only a subset selected by attribute values.
- Consider what happens to the excluded instances and if these should be displayed or not. (Note that the model allows replacement of one feature type by multiple feature subsets, for example a Chart layer Ice Area may be replaced by multiple ice features from a Sea Ice layer).
- Consider whether the spatial attributes change and implications if they do. For example, does the replacing product include features of both point and surface while the replaced product includes only point features.

16-D-2.10 Element S100_IC_SuppressedFeatureInstance

The **S100_IC_SuppressedFeatureInstance** element is used by **S100_IC_PredefinedCombination** for selecting instances from one product to be replaced by instances from another product.

The attributes *primaryProduct*, *primaryFeatureCode* and *primarySelector* specify the primary feature and product, which is to be suppressed.

The attributes *secondaryProduct*, *secondaryFeatureCode* and *secondarySelector* specify the secondary feature and product, which should replace the suppressed instance(s).

The attributes *outputProduct* and *outputFeatureCode* specify the replacement feature and product, and should be identical to the secondary product. The output feature does not need to have any hybrid characteristics, see clause 16-4.4.2.7.

The reference to a **S100_IC_Feature** gives the resulting feature’s display specification in the display planes section of the Interoperability Catalogue.

The optional reference to a S100_IC_SimpleRule gives the rules describing how a set of feature instances are combined to create a hybrid feature type.

Where two different set of features instances are to be suppressed by one common feature instance, two instances of S100_IC_SuppressedFeatureInstance is needed.

<u>Distinction:</u> S100_IC_HybridFeature.

16-D-2.11 Element S100_IC_HybridFeature

Complex operations, including conversion of input feature instances into new features (hybridization) are described by associated **S100_IC_HybridFeature** elements. The replacement and hybridization rules are described in Clause 16-4.

The attributes *primaryProduct*, *primaryFeatureCode* and *primarySelector* specify the primary inputs that will be suppressed from rendering.

The attributes *secondaryProduct*, *secondaryFeatureCode* and *secondarySelector* specify the secondary inputs that will be suppressed from rendering.

The attributes *outputProduct* and *outputFeatureCode* specify The attributes *outputProduct* and *outputFeatureCode* specify the replacement feature and product, which may be defined in a hybrid FC and hybrid PC.

The reference to a **S100_IC_Feature** gives the resulting feature's display specification in the display planes section of the Interoperability Catalogue.

The reference to either a **S100_IC_ThematicRule** or a **S100_IC_CompleteRule** gives the rules describing how a set of feature instances are combined to create a hybrid feature type.

Distinction: S100_IC_SuppressedFeatureInstance.

16-D-2.12 Element S100_IC_CompleteRule

This class is used for rules describing how a set of feature instances is combined to create a hybrid feature type. Complete rules may operate on both thematic and spatial attributes.

16-D-2.13 Element S100_IC_ThematicRule

This class is used for rules describing how a set of feature instances is combined to create a hybrid feature type. Thematic rules may operate on only thematic attributes. The input features are required to have spatially equal geometry within a tolerance set by the system.

16-D-2.14 Element S100_IC_SimpleRule

This class is used for rules describing how a set of feature instances is combined to create a hybrid feature type. Simple rules may operate only on thematic attributes. Simple rules treat thematic attributes uniformly during hybridization and therefore do not mention specific attributes. Location/extent spatial attributes of all input features must be spatially equal.

16-D-2.15 Element S100_IC_HybridFC

Class for holding a reference to a Feature Catalogue defining any "hybrid feature types" that are created by combining feature types from two or more products for the purposes of an interoperable display. The Feature Catalogue must conform to S-100 Part 5.

16-D-2.16 Element S100_IC_HybridPC

Class for holding a reference to a Portrayal Catalogue defining portrayal rules for the "hybrid feature types" defined in a Hybrid Features Catalogue. The Portrayal Catalogue must conform to S-100 Part 9.

Appendix 16-E
Implementation Guidance
(informative)

16-E-1 Implementation guidance

This appendix provides guidance on implementation issues that should be considered when developing an Interoperability Catalogue for an end-user system. Individual interoperability specifications may elaborate and modify this guidance as appropriate.

16-E-2 Reduce demand on user attention – display adjustment

It is recommended that implementers of end user systems supporting Interoperability Catalogues include decluttering techniques, such as minimizing overlaps of both symbols and text and minimization of the number of colours simultaneously on the display.

16-E-3 Reduce demand on user attention – avoid text overload

It is recommended that implementers of end user systems provide means for text notes to be shown in a manner that limits the obscuring other for example by including for separate text display from graphic display.

It is recommended that means are provided for limiting the amount of text shown simultaneously both in-graphic, over-graphic, and in a separate auxiliary display.

Where possible it is recommended that in-line text is kept shorter than text from a support file. This may be done through limiting the allowed text in some attributes in Product Specifications. For example, allow as many as 300 characters in any attribute intended for in-line text.

Interoperability Catalogue developers should review text handing of in scope Product Specifications, including what individual Data Classification and Encoding Guides say and what Portrayal Catalogues do with text attributes. This review should be done in collaboration with Product Specifications developers, since they can be expected to know which attributes can be expected to contain long text and which contain short text.

16-E-4 Support for novice users

It is recommended that end user implementations are permitted to have "novice" modes or user interface controls, which provide shortcuts for inexperienced users.

16-E-5 Reduce demand on user attention – planning and monitoring modes in navigational systems

Navigational systems with planning mode should be permitted to provide more powerful information search or processing functionality at the expense of more user attention.

Navigational systems with route monitoring mode should provide means for showing the information required for route monitoring while allowing users to also focus on other tasks.

16-E-6 Interoperability and data coverage

The interoperability rules and interleaving operations described in an Interoperability Catalogue should only apply in areas where the products referenced in the rule or interleaving operation have data coverage at the current display scale on the navigation system.

If data coverage for some of the products in the selected predefined combination is absent in an area, the rules and interleaving operations referring to products which do have data coverage in the area in question should continue to apply in that area. Rules and interleaving operations referring to products which do not have data coverage in the area should not apply in the area in question.

Implementations in end user systems should be capable of indicating parts of the display screen where (a) interoperability is partially applicable because some of the data products in a predefined combination do not have data coverage while others do have coverage; (b) interoperability is not applicable at all because the data products in the selected predefined combination do not have coverage.

> Note: Depiction and symbols for such distinguished parts of the screen may be a matter for Performance Standards but an off-graphic message on the system, or an adaptation of appropriate warning symbology may be suitable.

16-E-7 Other significant information

The inclusion in Interoperability Catalogues of data products whose interoperability has not been discussed with the relevant Product Specification development team is recommended against.

There should be a dialogue between interoperability teams and Product Specification teams, so that new changes to Product Specifications are ensured to be covered by Interoperability Catalogues.

Feature Catalogue and Portrayal Catalogue development teams should be considered stakeholders for hybrid catalogues.

16-E-8 Phased implementation

Implementation of interoperability may be done in phases and this can be done by utilizing the interoperability level attributes (see 16-8) to segment different functionalities of an Interoperability Catalogue.

S-100 – Part 16a

Harmonised Portrayal of S-100 Products

Page intentionally left blank

Contents

Page intentionally left blank

16a-1 Scope

This Part specifies the principles for harmonising portrayal and other presentational functionalities across different S-100 based data products for the purpose of improving the user experience and reducing ambiguities within systems utilising multiple S-100 based data products. It also describes the relevant International Maritime Organization (IMO) guidance and resources within International Hydrographic Organization (IHO) that support efforts in portrayal harmonisation. It does not address the portrayal process, functionality, or architecture, which are addressed in other S-100 Parts (especially 9 and 9A), but instead focuses on presentational design aspects, such as display organisation, colours, and symbology.

16a-2 Conformance

This Part is not a profile of an ISO or other standard. Instead, it is based on – and, where possible, conforms to – specifications for the portrayal of Electronic Nautical Charts (ENCs) and general principles for ergonomics and Human-Computer Interaction (HCI). Much of this document is derived from the IHO S-52 Standard, especially clauses 2, 3 and 4.

This Part is based on specifications that evolved from studies and discussions by IHO, IMO, International Electrotechnical Commission (IEC), and manufacturers. These efforts resulted in a set of related standards and specifications promulgated by IHO, IMO, and IEC, which are cited in the references section of this Part and summarised in clause 16a-4.1. The general principles for ergonomics and human-computer interaction (HCI) have been described in various HCI and ergonomics publications.

16a-3 References

16a-3.1 Normative references

A.1021(26) *Code on Alerts and Indicators (2009), IMO Resolution A.1021(26)*, 2009.

IEC 61174 *Maritime navigation and radiocommunication equipment and systems – Electronic chart display and information system (ECDIS) – Operational and performance requirements, methods of testing and required test results*. International Electrotechnical Commission (IEC), Edition 4.0, 2015.

IEC 62288 *Maritime navigation and radiocommunication equipment and systems – Presentation of navigation-related information on shipborne navigational displays – General requirements, methods of testing and required test results*. International Electrotechnical Commission (IEC), Edition 2.0, 2014.

MSC.191(79) *Amendments to the Performance Standards for the Presentation of Navigation-Related Information on Shipborne Navigational Displays, IMO Resolution MSC.191(79), 2004*. As amended by MSC 466(101).

MSC.232(82) *Revised Performance Standards for Electronic Chart Display And Information Systems (ECDIS), IMO Resolution MSC.232(82)*, 2006.

MSC.252(83) *Performance Standards for Integrated Navigation Systems (INS), IMO Resolution 252(83), 2007*.

MSC.302(87) *Adoption of Performance Standards for Bridge Alert Management, IMO Resolution 302(87), 2010*.

MSC.466(101) *Amendments to the Performance Standards for the Presentation of Navigation-Related Information on Shipborne Navigational Displays (Resolution MSC.191(79)), 2019*.

MSC.1593 *Interim Guidelines for the Harmonized Display of Navigation Information Received via Communication Equipment, IMO MSC.1/Circ.1593, 2018*.

MSC.1609 *Guidelines for the Standardization of User Interface Design for Navigation Equipment, IMO MSC.1/Circ.1609, 2019*.

SOLAS V *Safety of Life at Sea, Chapter V, Safety of Navigation, SOLAS Chapter V, 2002*.

SN.243/2 *Guidelines for the Presentation of Navigational-Related Symbols, Terms and Abbreviations, IMO SN.1/Circ.243/Rev.2, 2019*.

16a-3.2 Informative references

CSS2 *Cascading Style Sheets, Level 2 (CSS2) Specification, W3C Recommendation 12-May-1998, REC-CSS2-19980512*, World Wide Web Consortium, May 1998. URL: http://www.w3.org/TR/1998/REC-CSS2-19980512 (retrieved 2020-08-12).

CSS2.1 *Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification, W3C Recommendation 07-June-2011, REC-CSS2-20110607*, World Wide Web Consortium, June 2011. URL: http://www.w3.org/TR/2011/REC-CSS2-20110607 (retrieved 2020-08-12).

IALA G1105 *Shore-side Portrayal Ensuring Harmonisation with E-Navigation Related Information*, International Association of Marine Aids to Navigation and Lighthouse Authorities, Edition 1, 2013.

IEC 60945 *Maritime navigation and radiocommunication equipment and systems – General requirements – Methods of testing and required test results*. International Electrotechnical Commission (IEC), Fourth Edition, 2002.

IEC 61924-2 *Maritime navigation and radiocommunication equipment and systems – Integrated navigation systems – Part 2: Modular structure for INS – Operational and performance requirements, methods of testing and required test results*. International Electrotechnical Commission (IEC), Edition 1.0, 2012.

IMPA-PPU *Guidelines on the Design and Use of Portable Pilot Units*, International Maritime Pilots' Association, 2016.

ISO 19117 *Geographic Information - Portrayal*. ISO Standard 19117 Edition 2, 2012.

RTCM 10900.6 *RTCM standard 10900.6 for Electronic Chart Systems (ECS)*, Radio Technical Commission for Maritime Services, July 2015.

S-4 IHO Publication S-4, *Regulations of the IHO for International (INT) Charts and Chart Specifications of the IHO*, Edition 4.8.0, October 2018. (Parts B and C in particular.)

S-52 IHO Publication S-52, *Specifications for Chart Content and Display Aspects of ECDIS*, Edition 6.1.1, June 2015.

16a-4 Introduction

In any system where several types of data products and sensor inputs will be used on a common screen, there is a need to harmonise their portrayal in the user interface. Harmonisation is needed to ensure users are not misled by, for example, contradictory symbology, text or colours. Harmonisation also is also needed reduce the risk of the screen becoming overloaded with information. Portrayal harmonisation is also needed to reduce the user burden in terms of training requirements and the quantity of visual variables which must be memorised and considered in assessing information displayed on screens, and to give improved predictability for system implementors. The IMO has developed guidance for presentation of e-Navigational related information and this guidance should be leveraged for any portrayal harmonisation framework.

This Part contains portrayal harmonisation that describes how S-100-based products should be used and displayed simultaneously. It is an overarching statement of principles which applies generally to all S-100-based Product Specifications. Since this does not depend on particular Product Specifications, flexibility is possible in the approach to designing and developing portrayal, and user interfaces, which should make it unnecessary to amend all involved Product Specifications should any additional Product Specifications be identified, or any portrayal rules need amendment.

One important aspect of this guidance is defining the classes of user system to which any guidance applies, as there are a great variety of potential users of S-100-based products. For purposes of portrayal harmonisation, this document divides applications and systems into the following broad categories (in practice often a matter of configuration and task rather than equipment or software):

- ECDIS, INS, and similar navigation display systems and applications - primarily used for voyage planning and monitoring, and collision avoidance, with ENC as primary geographical data; covered by SOLAS and require type approval for meeting carriage requirements;
- Electronic Charting Systems (ECS) and similar systems and applications (for example, Portable Pilot Units (PPU)) - primarily used for route monitoring and voyage planning, with ENC or equivalents as the primary data; type approval not required;
- Other types of S-100 based user systems and applications, including shore systems.

Most S-100 based Product Specifications to date have been developed for ECDIS. ECDIS is a long-established and well documented concept, and since other users of S-100-based products are likely to know how the ECDIS concept relates to their systems, ECDIS is therefore the best-known user system upon which to model the portrayal harmonisation guidance.

Given the importance of ECDIS and other navigation displays (for route monitoring or voyage planning) to marine transport, this Part defines principles focused on such navigation displays as well as more general principles intended for other application domains.

16a-4.1 Organisations and standards

Several organisations are involved in developing standards and guidelines for portrayal, alerts and indications, and other user interface elements. Figure 16a-1 below depicts the main organisations which contribute to the development of user interface standards and other specifications, and the main influences between different kinds of standards/specifications. The current set of standards and specifications relevant to portrayal and the user experience can be characterised as follows:

- Presentation, performance, and user experience standards for ECDIS and INS are generally developed under IMO control, and include standards and guidelines for display and user interaction, including alerts. Specification of navigation-related symbols, including chart symbols, have generally been assigned to IHO as the subject matter experts.
- The framework for data content and display is defined in IHO standards, primarily in the form of the S-100 standard, supplemented by the IHO Geospatial Information (GI) registry as a repository of data object concepts, type definitions, and symbology. S-100 also provides an abstract specification for visual interoperability for displaying multiple products on the same screen; the only concrete specification for interoperability at the time of writing is S-98, which covers EDCIS interoperability.
- Specifications for individual data products are produced by several organisations depending on the specific domain. Standards covering geographic data and marine services are developed by IHO, weather and ice-related Product Specifications is generally under WMO responsibility, navigational warnings are an IHO/IMO joint effort by the WWNWS sub-committee, specifications for Inland ENCs are produced by the Inland ENC Harmonisation Group (IEHG), and IEC manages a standard for route exchange[1].
- A Product Specification should include the symbols and portrayal rules for the data product, however where this is not included, the system can specify the portrayal rules for the resulting products. System specified portrayal rules should also follow the portrayal harmonisation guidance.
- Standards for testing operation and performance of equipment and systems are developed under IEC control.

The various categories of standards and specifications are generally interrelated and often co-dependent. IMO navigation and safety standards refer to IHO symbology for representing geographic features, IHO symbology is defined taking into account user interface guidelines in IMO standards, and

[1] Other organisations are also developing S-100-based standards for information exchange, of varying relevance to portrayal and user interaction, shipboard or onshore. The list of Product Specifications in Figure 16a-1 is representative, not comprehensive.

the IEC operational, performance, and testing standards influence and are influenced by both IMO and IHO documents. Individual Product Specifications are based on the S-100 framework and are referenced by testing guidelines. Product Specifications are based on the S-100 framework and the features and attributes in their data models, as well as the symbols and colours they specify for portrayal of the data, should all be in registered in the IHO GI Registry. These interrelationships are depicted in Figure 16a-1.

Developers must conform to the mandatory requirements of the particular standards which apply to an application or system. In case of a conflict between this Part and a mandatory requirement in an applicable cited standard, the requirement in the standard supersedes the guidance in this Part.

Summaries of the various standards and specifications are provided in Appendix A.

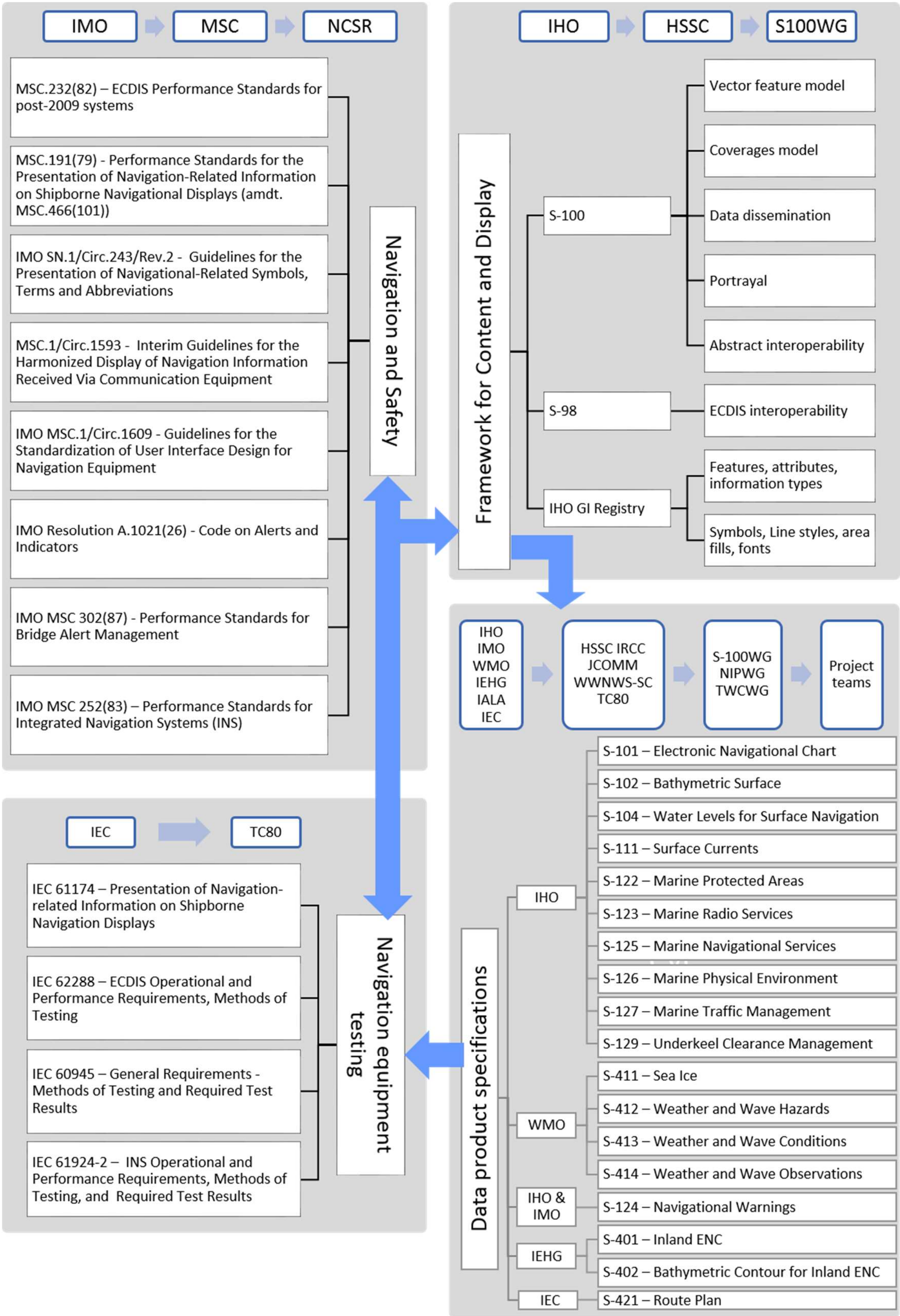


Figure 16a-1 – Overview of organisations, specifications, and user interface standards

16a-4.2 Application systems concepts and limitations

This clause describes the concept and limitations of the main types of application systems covered by this Part. ECDIS and INS are covered in S-98 and the focus of this clause is on considerations that affect content and behaviour of other systems.

16a-4.2.1 Electronic Chart Systems (ECS)

ECS is a common denominator for all systems that portray electronic charts. RTCM 10900.6 defines Electronic Chart System as an electronic navigation system which complies with the requirements it sets or in IEC 62376. ECS systems may meet the ECDIS requirements, have additional functionality or be a simple tablet or phone with a GNSS function and capabilty to show electronic charts. RTCM 10900.6 gives a grading of ECS systems from Class A through D; where Class A is equal to an ECDIS backup and Class D is any device “intended to plot the position of ships that do not operate offshore. They are not required to have all of the functionality of a Class C ECS. They are required to display electronic chart information and plot a ship’s position, but are not required to display eMSI, or to monitor the ship’s position or to provide voyage planning or voyage monitoring functionality”.

The graphical appearance of charted areas on the screen is usually similar to that of an ECDIS, with elements (for example, borders or off-graphic indicators) added. These systems will generally also have additional data products loaded, for example official products containing Marine Protected Areas or Marine Radio Services, or unofficial products such as points of interest (POI) to leisure craft users.

ECS may provide functionality not available on ECDIS and while ECS may not be as constrained as ECDIS, such systems, if intended for shipboard use (for example, Portable Pilot Units (PPU)), should, as far as practicable, follow the same principles for symbology and colours. Their text display elements, display organisations, pick reports, and user interfaces may be more elaborate than ECDIS (so as to allow users more access to information or more detailed information), but should have compatible structure and functionality (for example, similar prominence for significant features, user interface labels and menu items should have the same meanings).

This principle is based on reducing, as far as possible, the cognitive switching penalty[2] when switching from planning to monitoring tasks on shipboard. The constraints in this clause are therefore less applicable to systems not intended for shipboard use (though *actual* use versus *intended* use must be considered).

16a-4.2.2 Other systems

Portrayal harmonisation is also beneficial in other systems, such as those that cover shore-side and specialised systems not intended for onboard route monitoring or ship movement monitoring or control, or other tasks allocated to an ECDIS, INS, or ECS. An example of such shore side systems is Vessel Traffic Management Information System (VTMIS). VTMIS can be regarded as an extension of a Vessel Traffic Service (VTS), since its main purpose is an Integrated Maritime Surveillance system. VTMIS incorporates other resources to allow associated services and other stakeholders to directly share VTS data or access to certain subsystems in order to increase the effectiveness of port or maritime activity operations as a whole, but may not relate to the purpose of the VTS itself.

VTMIS are generally specified individually to the specific VTS but may be governed by national or regional regulations. For example, Directive 2002/59/EC of the European Parliament governs VTS inspections of European Maritime Safety Authority (EMSA) member countries. IALA provides guidance about portrayal of data and information in shore-side systems, including VTMIS (for example, G1105 – Shore-side Portrayal Ensuring Harmonisation with E-Navigation Related Information).

Generally, ENC charts form the base layer in a VTMIS and other data sources form layers of additional information or enhanced information to aid the VTMIS operator in monitoring and decision making. Examples of additional layers included radar and AIS for target tracking, AIS-ASM (AIS Application Specific Messages) for sensor information such as met-hydro information, air gaps and special zones such as irregular speed zones. Moreover, radio services may form an integral layer for service assistance. Other examples include MSI such as navigational warnings and meteorological warnings. Oceanographic information such as surface currents and water level may also be important layers.

---

[2] See, for example, “Multitasking: Switching costs”, American Psychological Association <https://www.apa.org/research/action/multitask>.

16a-5 General principles

IMO Guidelines for the Presentation of Navigational-Related Symbols, Terms and Abbreviations (SN.1/Circ.243/Rev.2) notes in section 15, the principles applied when designing the appendices to the guidelines. Not all these principles are directly applicable to harmonisation of portrayal between data products as they are mainly applicable for Human Machine Interfaces. However, even the principles that are not directly applicable still provides important indirect guidance that can positively influence portrayal harmonisation. Particularly noteworthy principles are:

- Consistency in use of symbols and patterns referencing Appendix 2 which provides information on icons, symbols and abbreviations that require standardization. Consistency enables recognition and detectability across the user interfaces of different navigation systems. Humans react positively to patterns and logical groups of items, and use categories to search for individual bits of information. User testing can identify groupings and patterns of information that should be prioritized for consistency. Patterns incorporate the way in which someone uses information and the types of information that are grouped together.
- Using location and grouping for consistency provides for recognition. Human perception and search work faster with cues than complete recall, especially when aided by consistency. The user must recognize where information is, or how to perform a process. In performing functions, the user should not need to recall where something is located, or the process for doing something. This is the ability for the user to recognize an event, process or information flow rather than recall the detail of how to get to that point. This is integral to usability.
- Prevent errors by ensuring users always see navigational critical information. Prevent errors, emergency exit – Continuous testing during development will identify possible error paths that can be removed. Users should be aware of how to navigate back to the start of a process, and also be aware of where they are in that process. The user should always be able to see navigation critical information even if layers of information are interwoven with the ENC/Radar.

Portrayal harmonisation must also consider if there should be a fixed set of layers that must be within the scope of the rules or if the rules should be more flexible to accommodate additional layers, such as future products or regional specialities. For example:

- A fixed list of layers gives predictability for implementers, since most variables are known and can be accounted for.
- A fixed list of layers is inflexible since new layers cannot be added without creating a new version of the rules and implementers may need to update user systems to add new layers.
- A flexible list of layers necessitates that the rules be more general and therefore easier to maintain, with less likelihood of having to create a new version of the rules for new layers. Also, users may create their own combinations and add layers as per need, such as in regional special circumstances.
- A flexible list of layers gives implementors less predictability of what layers to expect with higher possibility of unexpected layer combinations. This necessitates more testing and more flexible user settings to accommodate user defined layer combinations, and avoid clutter that impact system usability.

Portrayal harmonisation guidelines apply to both vector and coverage type data as far as practical.

16a-5.1 Symbol harmonisation

Symbols are used to visually convey information and when utilized in the same environment, for example, marine, need to convey consistent information regardless of which product they are used in. As noted in IMO SN.1/Circ.243/Rev.2, “consistency enables recognition and detectability across the user interfaces of different navigation systems. Humans react positively to patterns and logical groups of items, and use categories to search for individual bits of information. User testing can identify groupings and patterns of information that should be prioritized for consistency. Patterns incorporate the way in which someone uses information and the types of information that are grouped together”. Symbols should therefore be harmonised for products used in the same general class of system/application to ensure the risks of contradictory messages are reduced as much as possible.

Symbols should be designed in accordance with the guidance within IMO SN.1/Circ.243/Rev.2 and IHO S-4. These two documents should be the first source of finding portrayal for features that need visualization in a navigation system. If suitable symbology is not found in either of these documents, the IHO Nautical Cartography Working Group (NCWG) is tasked with coordinating portrayal for IHO related products and may be able to offer assistance. Given IHO's leadership in defining the portrayal of the chart, all organizations that produce data products intended for the navigation screen or related systems should coordinate the development of any new symbology with IHO NCWG.

New symbology should be tested within the product itself, and also in all expected product combinations to ensure sufficient harmonisation and that any risk of ambiguities is sufficiently addressed.

16a-5.2 Visual interoperability

The Interoperability Catalogue Model, see Part 16, is a framework that can be used to create a catalogue of rules for how to combine a set of layers. This set can be a fixed list or a flexible approach.

The S-100 Product Specifications may have only considered portrayal harmonisation with ENC, is also a S-100 Product Specification. Therefore, when several layers are visualized simultaneously there is a risk of clutter that may obscure significant features and increase the risk of unsafe operation. The Interoperability Catalogue Model framework should be utilized to limit the number of simultaneous on-screen layers to a logical set, for example by using the preconfigured combination function in normal operating mode, whilst permitting more flexibility for users with specialized requirements. Logical sets can be viewed as those layers useful in a particular operation. Within a logical set significant features should be given priority. Significant features may be regarded as those features that must not be turned off at any moment during important operations, such as route monitoring in navigation systems. In ECDIS and ENC significant features are considered to be in the display base. In other S-100 based products and systems there may also be significant features and these should be considered in the same spirit as the display base defined in the IMO ECDIS performance standard (Resolution MSC.232(82)). Great care should be taken when combining layers to minimise the risk of obscuring any significant feature. Considerations should be given to making the use of the pre-defined combinations of Product Specifications concept to limit the possible combinations, and to assign viewing group layers to specific pre-defined combinations of Product Specifications.

16a-5.3 De-cluttering strategies

All S-100 based Product Specifications that include portrayal need to consider decluttering strategies. This is necessary to reduce the risk of clutter when conforming data products are utilized in user systems either in isolation or as a layer within a larger ecosystem. Any S-100 based products used in isolation may cause screen clutter in a system when used at a scale where the screen density of the data content start interfering with each other. Similarly, when S-100 based data products are used as layers in a system, the data density may be such that the content start interfering with each other due to quantity of content even at larger scales due to quantity of content. Either scenario needs to be considered by Product Specification developers and system implementers; and mitigation strategies need to be employed.

One clutter mitigation strategy that can be employed is to introduce scale bands in the products to limit data density at the various scale ranges by generalization at lower scales, including thinning for raster data, while including the greatest detail at the largest scales. This method works best on the individual product series as it cannot account for impacts of any other layers, but adding scale bands can be part of an overall clutter mitigation strategy for the most feature dense products.

A second mitigation strategy can be adding scale minimum and scale maximum attributes to portrayed feature classes to turn off less significant information at either small scales or large scales. The scale minimum attribute is used as a limit to indicate the scale at which portrayal of a feature is turned off as the screen is scaled to smaller scales, for example by turning off point symbols that start interfering with each other due to density at smaller scales. The scale maximum attribute is used to turn off the feature portrayal at large scales, for example large areas that may be considered of less significance at large scales and can cause clutter, for example by area patterns. This method works best on the individual product series as it cannot account for impacts of any other layers, but can be part of an overall clutter mitigation strategy. Producers should ensure consistency between products.

Configuring an interoperability catalogue can be a third mitigation strategy by defining rules for how predefined combinations of products should interact, by for example turning off less significant features in one product to enable easier identification of more significant features in other products. Interoperability catalogues can also be used to define rules for combining data from different products into hybrid features that can eliminate the need to portray multiple copies of the original data. Utilization of an interoperability catalogue is most suitable for situations where multiple layers interact and are at risk of causing screen clutter.

These three strategies are not mutually exclusive and any combination of them can be utilised as well as in combination with other methods for clutter mitigation. Additional strategies are discussed in the subsequent sections.

16a-6 Display Organisation and Operation

Some systems have detailed specifications and requirments established; for example, ECDIS. Other systems may have less strict rules and requirements but may follow similar patterns in full or in parts of other performance standards. Examples of such systems can be Portable Pilot Units (PPU) which may follow some of the ECDIS performance standards because they are both used in the same environment and utilize much of the same data, additionally PPUs may have local adoptations to the regions they are utlized in. In general, the parts of portrayal guidance that relate to display organization and system operation are applicable to all S-100 based Product Specifications regardless of the intended systems that will use the S-100 based data that derive from these Product Specifications (though systems may provide elaborations on basic organisations, as outlined in clause 16a-4.2.2).

16a-6.1 Operating modes

Some systems (for example, ECDIS) are designed with different colour modes to account for ambient light conditions on the bridge of the vessel. These modes support an operating environment that preserves the navigator's light sensitive vision which can be critical in spotting situations that impact safety of navigation, especially at night. All S-100 based Product Specifications that are intended for navigation screens should support the operating modes of the navigation system where the data product is intended to be used. Note that IMO/IEC performance requirements for certain systems, such as ECDIS, may require such support. Manufacturers and users of S-100 products whose Portrayal Catalogue does not have a suitable colour mode for any combination of system type and lighting conditions are invited to discuss the matter with IHO.

The IHO ENC Standards Maintenance Working Group (ENCWG) is tasked with managing S-52 - Specifications for Chart Content and Display Aspects of ECDIS which specify the portrayal rules for S-57 ENCs. The Nautical Cartography Working Group is tasked with coordinating portrayal. Jointly these two working groups may be a good source for advice on colour usage in different portrayal operating modes.

Given IHO's leadership in defining the portrayal of the nautical chart with the Nautical Cartography Working Group (NCWG), all organizations that produce data products intended for the navigation screen should coordinate their colour usage with IHO. Recommendations for colour usage in different portrayal operating modes on ECDIS will be published in an Annex to the IHO ECDIS interoperability specification.

16a-7 Colours

Colours and combinations of colours often have specific meanings associated to them; this is also true for navigation displays. For example, on a chart black usually means a physical object or boundary (for example, coast line), while magenta usually means a non-physical object or boundary (for example, restricted area boundary). Other examples include buoyage systems marks where combinations of yellow and black bands are used to indicate a cardinal direction from a known danger.

Another consideration with colour choices is that any symbology using red may be an issue for any system operated in night conditions as these symbols may become very difficult to distinguish in red ambient light often used to preserve night vision. Therefore the use of red should be avoided where possible and always used with a symbol shape that improves recognition. Adding text can also reduce risk of mix-ups.

With so many implications on colour choices for portrayal, it is important to ensure that the use of colour is harmonised between layers and systems to reduce the risk of giving the user ambiguous information. Product Specifications should attempt to use existing colours registered within the IHO Portrayal Registry.

Given IHO's experience and leadership in defining the portrayal of the chart, all organizations that produce data products intended for the navigation screen should coordinate their colour usage with IHO. Within IHO the Nautical Cartography Working Group is tasked with coordinating portrayal.

16a-7.1 Colour assignment

Colour assignments for ECDIS, INS, and any similar navigation systems covered by IMO Performance Standards must conform to S-98 Colour Assignments.

Colour assignments for bridge or charthouse systems not covered by IMO Performance Standards for navigation systems should conform to S-98 Colour Assignments, but may add other colour assignments as necessary for their purposes.

Colour assignments for other systems may conform to S-98 Colour Assignments, but may depart from or extend it, depending on application requirements or user environments.

16a-7.2 Colour tokens, profiles and palettes

S-100 based Product Specifications use the S-100 portrayal concept which consists of several components that can be registered in the GI Registry and may therefore be shared among different Product Specifications. Colour tokens and colour profiles are examples of such shareable concepts. The colour tokens are used in Portrayal Catalogues to specify the particular variation of a general colour using an address in sRGB and/or CIE colour space coordinates. Historically, several colour tokens have been assigned to specific usages within ENC for concepts such as depth areas, land areas, regulated areas, buoys, lights, etc. These feature concepts may be reused in any S-100 based Product Specifications, which may also add their own colour tokens. This necessitates harmonisation of the use of colour tokens. When a feature concept is portrayed by two or more Product Specifications it is recommended that the same colour tokens are used in all Portrayal Catalogues.

16a-8 Text

IMO MSC.191(79), as amended, requires in section 5.2.3 that text be presented using simple unambiguous language that is easy to understand. Navigation terms and abbreviations should be presented using the nomenclature defined in the Guidelines and encouraged their use for all shipborne navigational systems and equipment (see SN.1/Circ.243, as revised and Appendix 2 of IMO MSC.1/Circ.1609). Similarly, S-4 (Section 500 - Text: Language, Numbers, Abbreviations, Names, Styles and Fonts) has defined common navigation terms and abbreviations that are used in the presentation of navigational information. This textual guidance should jointly with the IMO guidance be used as a reference for harmonising the use of text in S-100 based Product Specifications intended for navigational use.

It is recommended that the guidance in IMO MSC.1/Circ.1609 be adhered to as it notes that when icons, terms and/or abbreviations are used, these must meet the requirements of the guideline. Moreover, it is noted that where a standard term, abbreviation, or icon is not available, another icon, term or abbreviation may be used, but these should not conflict with those listed in the aforementioned guideline. Therefore, where terms or abbreviations are not available in IMO MSC.1/Circ.1609; IHO S-4 and IMO SN.1/Circ.243, as revised, should be examined for suitable alternatives. Only when these guidelines have been examined and found to not contain the suitable terms or abbreviations, should considerations be given to creating new terms or abbreviations. If new terms or abbreviations are created after due consideration of the above mentioned recommendations, efforts should be made to add the new terms or abbreviations to the suitable guideline to ensure harmonisation.

While the above mentioned IMO instruments are specifically intended for ECDIS, other classes of systems or applications intended for use in the marine context should use the same terms and abbreviations.

16a-9 Pick reports

IMO Circular 1609 notes that “large variations in the user interfaces of electronic equipment can significantly inhibit an operator's effectiveness in performing navigational tasks. Where there is significant variation in buttons, icons, actions, workflows, processes, units of measure or location of information, there is a commensurate increase in the time required for equipment familiarization and the risk of operational error, particularly in challenging navigational situations”. Pick reports have historically been defined for the individual Product Specification or by the implementor, leaving room for ambiguity that has led to difficulties for users. Additional challenges for the user may occur when two or more products that are being used simultaneously have conflicting configurations for pick reports. Similar complications can also be encountered when different systems in use have significantly different user interfaces for the same type of data. This would also require additional training to familiarize users with products that can be safety critical. Harmonised portrayal pick reports should therefore be a goal for any system that will use S-100 based products.

Pick reports should allow access to information from all visible/enabled underlying products. Data should be organized to facilitate navigation through complex reports in a manner that is logical when considering the layer order displayed on the screen. Human Machine Interface (HMI) strategies should be well-thought-out to make the task of reviewing the pick report requiring as little effort as possible for the user. This can be facilitated by harmonised look and behaviour across products and systems when used in the same operational mode or same context. This implies that Product Specifications remain flexible on how pick reports are designed. Flexibility should also be given to permit some variation between operation modes where appropriate. Pick reports should have a sort order that reflects the priority of the products on the screen.

16a-10 Alerts and Indications

All Product Specifications that are intended for a navigation context should specify any feature combinations that match one or more of the areas for which alert or indication should be given to ensure there is a harmonised implementation in user systems. Such specification could be done using a machine-readable alerts and indication catalogue, but caution is needed as this catalogue will likely be developed for one specific system, such as ECDIS. Therefore, implementers may find that some of the content of an alerts and indication catalogue is not applicable for a non-ECDIS system and only a subset should be utilized. This may also necessitate development of additional system specific alerts or indications.

IMO Resolution MSC.232(82) states in 11.3 and 11.4 and their sub paragraphs how an ECDIS should respond to risk of crossing, dangers, prohibited areas or areas with special conditions. In Appendix 4 and Appendix 5 of the same resolution details of which areas ECDIS should detect and provide an alert or indication for are given. This guidance is sufficiently high level that it should be useful for most S-100 based products and other relevant systems, and should therefore be consulted when designing alerts and indications.

Within IHO the ENC Standards Maintenance Working Group (ENCWG) is tasked with managing S-52 - Specifications for Chart Content and Display Aspects of ECDIS which specify the portrayal rules for S-57 ENC. This includes guidance for which feature attribute combinations should be considered to match the IMO ECDIS Performance Rules on alerts and indications. Given IHO’s leadership in defining the portrayal of the chart, and the interdependence of portrayal with other safety critical behaviour of systems, all organizations that produce data products intended for the navigation screen should coordinate their alert and indication rules with IHO.

16a-11 Portrayal harmonisation between new versions of Product Specifications

S-100 based Product Specifications are largely maintained using the S-100 maintenance regime, which classifies revised versions as clarifications, corrections, or new editions. These categories correspond to increasing significance and complexity. New functions are generally recommended to be in a new edition. This means that new portrayal elements, such as new symbols, may require a new edition of a Product Specification. The reasons behind such new symbology can be varied, such as changes to IMO guidelines, new user requirements or new technology. When new editions are required, it is important to carefully consider the impact of any portrayal change, as significant changes may have major consequences on equipment and users. If there is proper justification for a significant change to portrayal between versions of a Product Specification, appropriate means should be taken to communicate these changes with sufficient time for users and systems to adapt. The standard lifecycle process detailed in IHO Resolution 2/2007, as amended, should be adhered to for IHO products; other organisations should either use the same process or adopt an equivalent process.

16a-12 Specifications for the display screen

This clause describes general requirements for display screens. More detailed requirements are provided in specific standards applicable for different classes of systems. For example, IMO MSC 232(82) specifies the requirements for ECDIS.

16a-12.1 Physical display requirements

The minimum effective size of the geographic portion of the display should be such that key information is generally visible without scrolling the display and user interaction directly with the geographic display, if any, is possible without requiring excessive user concentration under typical conditions.

The details of how this principle applies to particular systems should be determined by analysing user tasks and user performance under expected and critical environmental conditions. Touch target sizes should take typical viewing distances into account.

Particular classes of systems may have minimum sizes set by controlling standards, for example, MSC 232(82) requires for ECDIS that the minimum effective size of the chart presentation for route monitoring is at least 270 mm × 270 mm.

16a-12.2 Colour display capability

Colour displays should be capable of at least 256 luminance steps in each of red, green and blue.

For night performance it is essential that the hardware has a graphics card capable of giving "blacker than black"; that is, complete control of colour, and that the software can control that function.

Displays must be capable of maintaining colour discrimination between symbols defined in the Portrayal Catalogues for the data products which may be used for tasks performed on the display. Typical users should be able to distinguish between colours which may be specified for the symbols they encounter in the performance of the task using the display.

Particular classes of systems may be subject to specific standards for colour conversion tolerances. For example, the ECDIS standards for colour conversion tolerances and colour calibration are described in the applicable ECDIS requirements, performance and test standards.

Specifications for particular classes of systems (in particular, ECDIS) may include a colour differentiation test diagram. This diagram is intended:

1) for use by the mariner to check and if necessary, re-adjust the controls, particularly for use at night;
2) for use by the mariner to verify that an ageing display remains capable of providing the necessary colour differentiation;
3) for initial colour verification of the day, dusk and night colour tables.

Appendix 16a-A
Organizations and Standards

16a-A-1 International Organizations and Standards

The IMO has issued recommendations and guidelines on how to present navigation related information. Of particular significance are the documents noted below.

- MSC.191(79), Recommendation on Performance Standards for the Presentation of Navigation-Related Information on Shipborne Navigational Displays, specifies the presentation of navigational information on the bridge of a ship, including the consistent use of navigational terms, abbreviations, colours and symbols, as well as other presentation characteristics. It also addresses the presentation of navigation information related to specific navigational tasks by recognizing the use of user selected presentations in addition to presentations required by the individual performance standards adopted by the Organization.
- IMO SN.1/Circ.243/Rev.2, Guidelines for the Presentation of Navigational-Related Symbols, Terms and Abbreviations, stems from a compelling user need for greater standardization to enhance usability across navigation equipment and systems. Significant variation between systems and equipment produced by different manufacturers has led to inconsistency in the way essential information is presented, understood and used to perform key navigation safety functions. Improved standardization of navigation systems will provide users with more timely access to essential information and functions that support safe navigation.

  MSC.191(79) and guidelines in SN.1/Circ.243/Rev.1 continue to apply as follows:

  - To radar equipment, electronic chart display and information system (ECDIS) and integrated navigation systems (INS) installed before 1 January 2024; and
  - To all other navigational displays on the bridge of a ship installed before 1 July 2025.

  MSC.191(79), as amended by resolution MSC.466(01), and guidelines in SN.1/Circ.243/Rev.2 are to be applied to equipment installed on or after the dates specified above[3].
- MSC.1/Circ.1593, Interim Guidelines for The Harmonized Display of Navigation Information Received Via Communication Equipment, provides interim guidelines for the display of navigation-related information received via communication equipment. It aims to ensure that information is displayed in an efficient, reliable and consistent format, in a manner that is easily interpreted to support decision-making. These Guidelines supplement the Performance standards for the presentation of navigation-related information on shipborne navigational displays (resolution MSC.191(79)) in regard to the presentation of navigation information received via communication equipment. The use of these Guidelines will ensure that navigation information received via communications equipment is displayed in a harmonised manner on the ships' navigational bridge.
- IMO MSC.1/Circ.1609, Guidelines for the Standardization of User Interface Design for Navigation Equipment, apply to Integrated Navigation Systems (INS), Electronic Chart Display and Information Systems (ECDIS) and radar equipment. They may also be applied to other electronic navigation equipment, and navigation sensors, where applicable, to improve standardization and usability. The aim of the Guidelines is to promote standardization of user interfaces to help meet user needs. The Guidelines have been developed in close collaboration with an international association of equipment manufacturers to ensure its efficient implementation. These Guidelines also aim to leave room for future innovation and development while still addressing the primary user need for standardization and usability. Improved standardization of the user interface and information used by seafarers to monitor, manage and perform navigational tasks will enhance situation awareness and safe and effective navigation.
- IMO Resolution A.1021(26), Code on Alerts and Indicators (2009), defines a classification of alerts (as emergency alarms, alarms, warnings, and cautions) and provides general design

[3] Anticipated dates are as of March 2020. Dates may be revised by IMO, and should be confirmed from the latest relevant IMO resolution/circular.

guidance and principles for achieving uniformity of type, location and priority for required alerts and indicators, as prescribed by SOLAS, other international conventions, shipping codes, and IMO performance standards and guidelines.

- IMO MSC 302(87), Performance Standards for Bridge Alert Management, describes performance standards for harmonising the priority, classification, handling, distribution and presentation of alerts.
- IMO MSC 252(83), Performance Standards for Integrated Navigation Systems (INS) describes requirements for the integration of navigational information, task-related requirements, alert management, and documentation requirements for an INS. These requirements supplement the performance standards for individual components. MSC 232(86) applies to equipment installed in 2011 and after; its predecessor, Annex 3 of MSC 86(70), applies to equipment installed from 2000 to 2010.

The IEC Technical Committee TC80 have developed standards for equipment, general and performance requirements, methods of testing, and symbols for selected non-geographic information (for example, ship activity, AIS, and radar information).

- IEC 61174 specifies the performance requirements, methods of testing and required test results of equipment conforming to IMO performance standards. This standard is based upon the performance standards of IMO resolution MSC.232(82). Reference is made, where appropriate, to IMO resolution MSC.232(82). It also includes relevant extracts from IHO publications (S-32, S-52, S-57, S-61, S-63 and S-64).
- IEC 62288 specifies the general requirements, methods of testing, and required test results, for the presentation of navigation-related information on shipborne navigational displays in support of IMO resolutions MSC.191(79) and MSC.302(87).
- IEC 60945 describes testing methods, operational tests and required test results for shipborne navigational equipment and electronic navigational aids. It is based on IMO Resolution A.694(17). IEC 60945 also covers shipborne radio equipment as well as addressing potential electromagnetic interference from other types of equipment.
- IEC 61924-2 describes requirements for the design, manufacture, integration, methods of testing and required test results for an INS to comply with the IMO requirements of MSC.252(83).

IHO standards and specifications describe data products, data content, data updating, and display, symbols and representations for geographic information, natural conditions, navigational hazards, and selected non-geographic information such as regulatory and administrative information.

ECS is a common denominator for all systems that portray electronic charts. RTCM 10900.6 defines Electronic Chart System as an electronic navigation system which complies with the requirements set forth in this standard or in IEC 62376. ECS systems may meet the ECDIS requirements (for example, backup ECDIS) or be a simple tablet with a GNSS function. RTCM 10900.6 gives a grading of ECS systems from Class A through D; where Class A is equal to an ECDIS backup and Class D is any device "intended to plot the position of ships that do not operate offshore. They are not required to have all of the functionality of a Class C ECS. They are required to display electronic chart information and plot a ship's position, but are not required to display eMSI, or to monitor the ship's position or to provide voyage planning or voyage monitoring functionality".

Portable Pilot Units (PPU) – The International Marine Pilots Association (IMPA) has issued a Guideline on the Design and Use of Portable Pilot Units. The guideline is high level, but stress that pilotage is a local mater and therefore no PPU solution can meet all needs in all pilotage areas. Therefore, PPUs are tailored to a particular pilotage area and there are a number of configurations that may be utilised.

Shore-side and specialised systems not intended for onboard route monitoring or ship movement monitoring or control, or other tasks allocated to an ECDIS, INS, or ECS see IALA G1105 – Shore-side Portrayal Ensuring Harmonisation with E-Navigation Related Information.

S-100 – Part 17

Discovery Metadata for Information Exchange Catalogues

Page intentionally left blank

Contents

Page intentionally left blank

17-1 Scope

The S-100 Discovery Metadata for Information Exchange Catalogues profile described in this part provides a specification for describing and creating Exchange Catalogues that enables users to identify, discover and manage content of the S-100 Exchange Sets. More importantly it leverages XML to allow machine to machine discovery and exchange of information about geographic datasets commonly produced by hydrographic organizations. Its purpose is the creation of metadata records that provide information about the identification, spatial and temporal extent, quality, Application Schema, spatial reference system, and distribution of digital geographic data. It is applicable to the cataloguing of datasets, clearinghouse activities, and the full description of geographic and non-geographic resources.

For information exchange, there are several categories of metadata required: metadata about the overall Exchange Catalogue, metadata about each of the datasets contained in the Catalogue, and metadata about the support files that make up the package. If the Exchange Catalogue contains any Feature, Portrayal or Interoperability Catalogues there is a provision to carry additional metadata about those.

This document is intended for developers and implementers of metadata applications, and provides a basic understanding of the principles and the overall requirements for standardisation of geographic information. It should be used in conjunction with the standards listed under clause 4a-4 – Normative references.

17-2 Conformance

17-2.1 Conformance of this Profile with other Standards

The S-100 Discovery Metadata for Information Exchange Catalogues profile adopts data types defined in other ISO standards, mainly in ISO 19115-1 Geographic information – Metadata – Part 1 - Fundamentals, ISO/TS 19115-3:2016, Geographic information - Metadata - XML Schema implementation for fundamental concepts and ISO 19136 Geographic Information - Geography Markup Language, along with their underlying obligations and conditions. The XML Schema references related to these external data types are embedded in the XML Schemas for this profile, thus ensuring the conformance of this profile with other standards during authoring and validation of XML catalogue instances.

17-2.2 Conformance to this Profile

The conformance to this profile can be confirmed by validating XML catalogue instances against the S-100 Metadata Profile Schemas which are available from the IHO S-100 repository.

17-3 Normative references

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies.

17-3.1 Profile definition

The following documents were the references used to define the S-100 Metadata Profile:

ISO 19115-1:2014, *Geographic information – Metadata – Part 1 - Fundamentals*

ISO 19115-1/Amdt01:2018, *Geographic information – Metadata – Part 1 - Fundamentals* (Amendment 1)

ISO 19115-2:2009, *Geographic information - Metadata - Part 2: Extensions for imagery and gridded data*

ISO 19119:2016, *Geographic information – Services*

ISO/TS 19115-3:2016, *Geographic information - Metadata - XML schema implementation for fundamental concepts*

17-3.2 Informative references

ISO 19115:2003, *Geographic information – Metadata*

ISO 19115:2003/Cor.1:2006, *Geographic information - Metadata* (Technical Corrigendum 1)

ISO/TS 19139:2007, *Geographic information - Metadata - XML schema implementation*

17-4 Overview

17-4.1 S-100 Exchange Set Structure

The S-100 Exchange Set is a data container that provides all the elements needed for a reliable and secure exchange of S-100 conformant data. It is intended to be a self-contained entity consisting of data files and metadata records packaged together using applicable data integrity and optional security provisions. The overall concept of the S-100 Exchange Set is a realization of the ISO 19115-3 classes, which fundamentally underpin the interchange of geospatial data and relevant metadata, as depicted in Figure 17-1 below. This Figure depicts, from left to right, the ISO data exchange structural classes, the relevant ISO classes for metadata for exchange, S-100 structure classes representing the S-100 Exchange Set components, and the relevant S-100 Exchange Set metadata classes.

Note that the S-100 structure classes represent components of the Exchange Set (files/folders/archives), not XML fragments in an exchange file. Accordingly, they do not have attributes nor do they have corresponding documentation tables in this Part.

Note also that the S-100 Exchange Set metadata classes are analogues of the corresponding ISO classes but (strictly speaking) not specialisations or realisations of them.

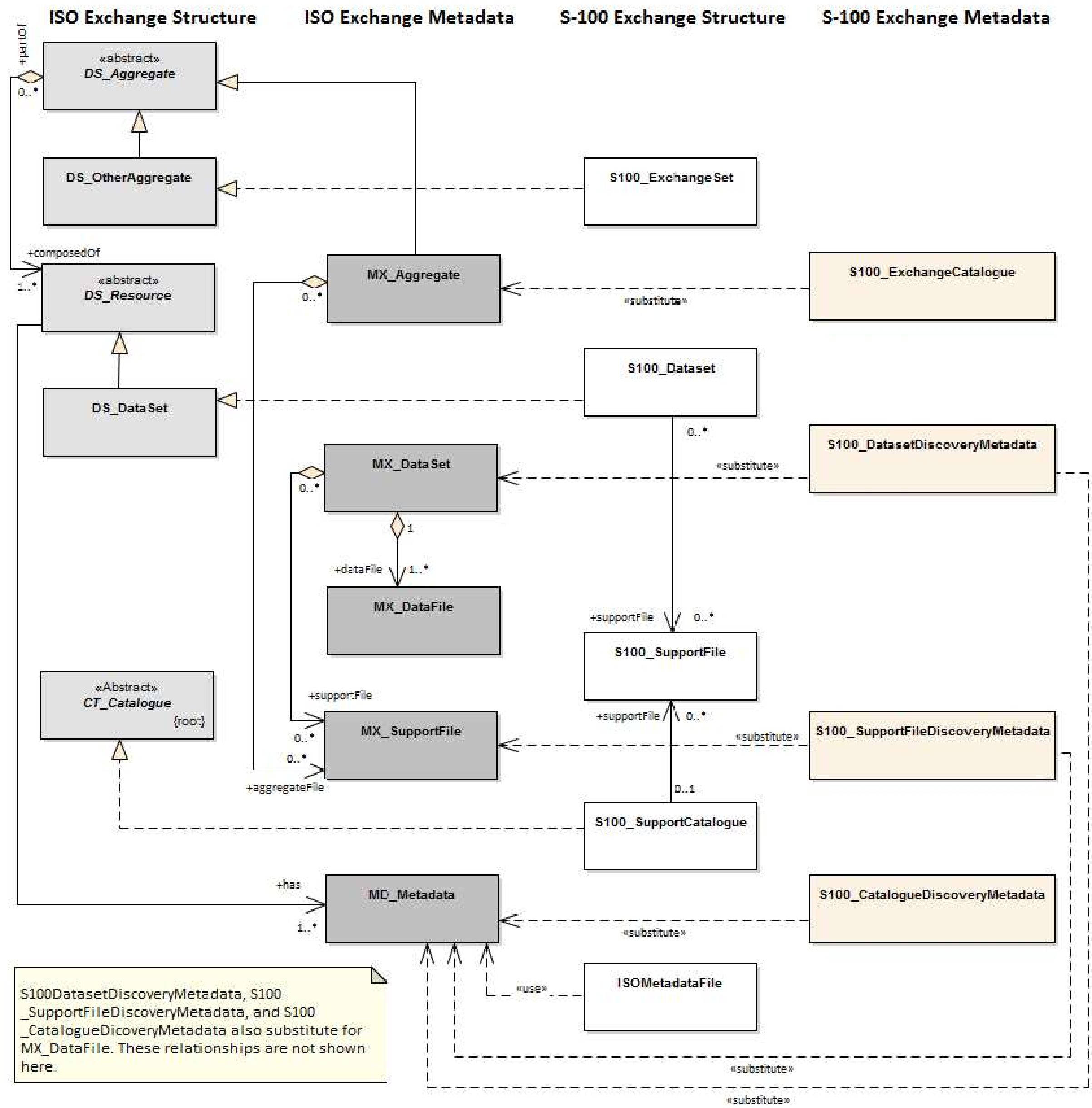


Figure 17-1 Realization of the Exchange Set classes

The above Figure illustrates the conceptual correspondence between data exchange provisions in ISO-19115-3 and S-100 standards. At an implementation level, the S-100 Exchange Set may include a combination of S-100 datasets, support files, and Catalogue files along with the metadata information for all such resources in the form of the S-100 Exchange Set Catalogue. Conceptually this leads to the more detailed model of the S-100 Exchange Set, as shown in Figure 17-2 below.

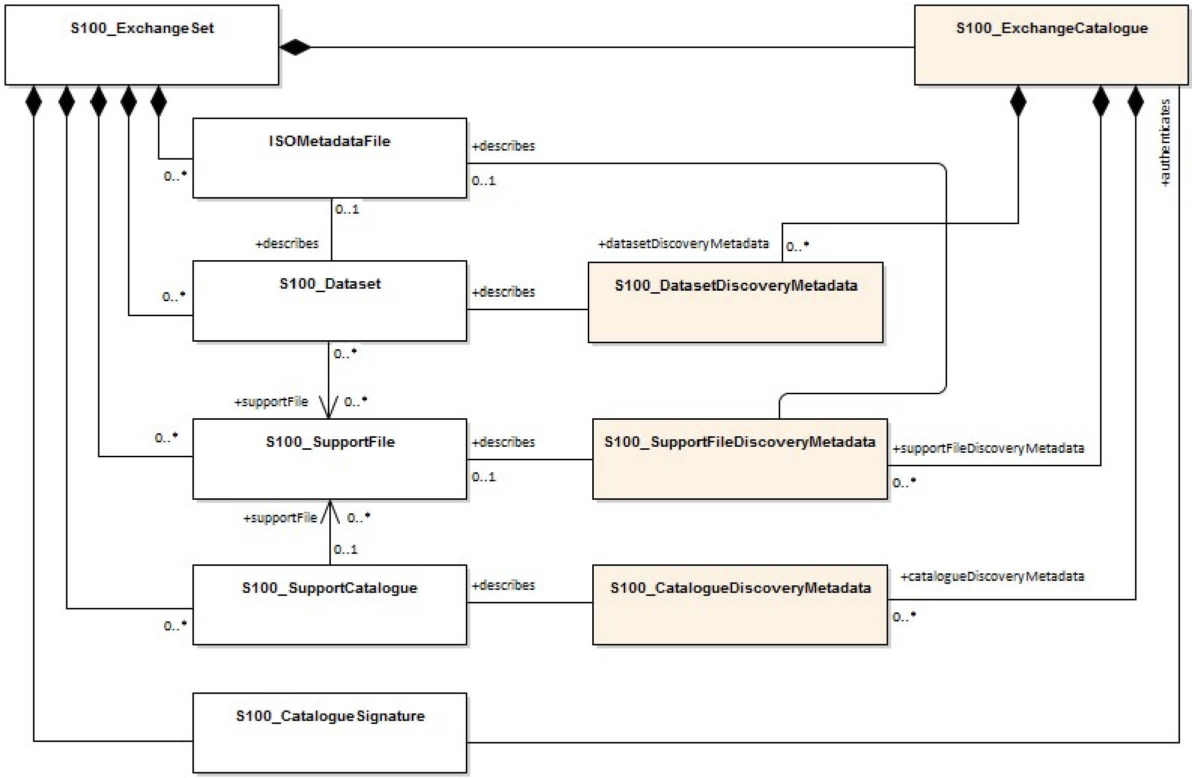


Figure 17-2 – S-100 Exchange Set

The conceptual model depicted in Figure 17-2 is very flexible and can be implemented in a variety of ways as virtually all components, except for the S-100_ExchangeCatalogue, are optional. This level of flexibility is essential to properly support the mainstream use case of exchanging geospatial data, as well as the use cases for releasing dataset and support file cancellation notices or new Catalogue releases without any data files present.

This approach ensures that an Exchange Set Catalogue is always included in any S-100 conformant Exchange Set, providing the essential discovery metadata about any included resources and their intended use.

Preparation of an Exchange Set consists of packaging its components using a predefined file folder structure shown in Figure 17-3 below.

17-4.2 S-100 Exchange Set Folder Structure.

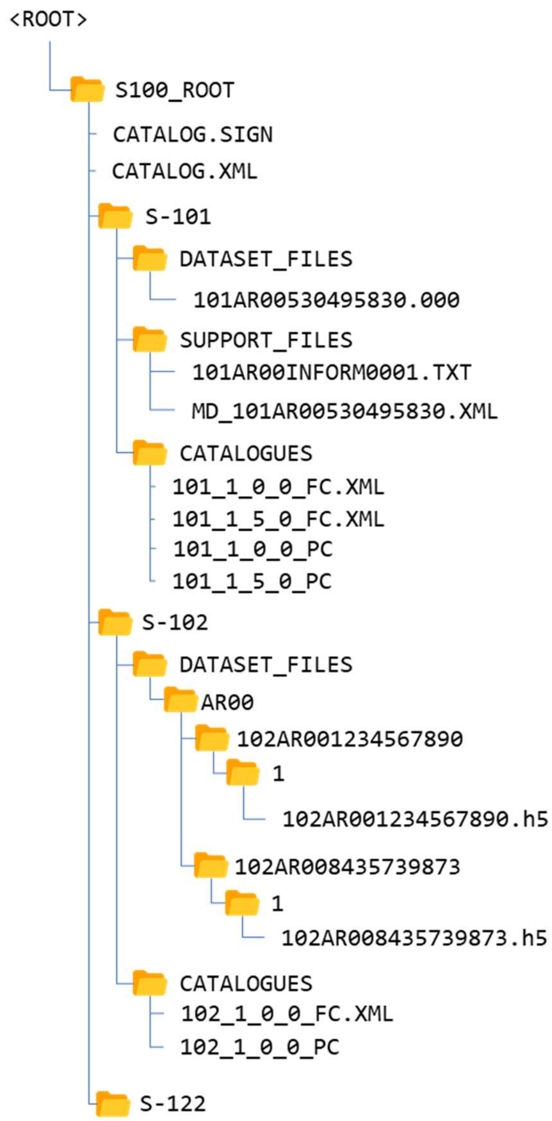


Figure 17-3 – An S-100 Exchange Set folder structure

1. An S-100 Exchange Set must contain an Exchange Set Catalogue, CATALOG.XML, its digital signature CATALOG.SIGN and may contain any number of S-100 conformant dataset files, support files and Catalogue files.
2. All S-100 content must be placed inside a top root folder named S100_ROOT. This is the only top level root folder in an Exchange Set containing only S-100 products.
3. The S100_ROOT folder must contain a subfolder for each specific S-100 Product Specification data type included in the Exchange Set; for example S-101, S-104, S-102 (names defined in the Product Specification Register of the IHO Geospatial Information (GI) Registry). These subfolders hold S-100 content specific to an individual Product Specification.
4. Each product subfolder must contain subfolders for the component dataset files (DATASET_FILES), support files (SUPPORT_FILES) and Catalogues (CATALOGUES) as required.
5. Individual data files may be optionally placed in their own subfolders (as demonstrated in the S-102 folder in Figure 17-3 above) or grouped together (as demonstrated in the S-101 folder in Figure 17-3 above). The ISOMetadataFile (see Figure 17-2) must be located in the SUPPORT_FILES folder. If used, all associated ISOMetadataFile must have unique names. The name of the associated XML Metadata file should not be used to describe the physical

content of the file. The associated XML Metadata file must be named MD_<data file base name>.XML.

6. Support files, on the other hand, may be grouped together in one folder to prevent duplication across multiple dataset folders. Similarly, when needed, an Exchange Set may carry additional Catalogues and/or their different versions which should also be grouped together in one folder.
7. The required Exchange Set Catalogue XML document instance must be named CATALOG.XML and placed in the S100_ROOT folder, together with its digital signature (CATALOG.SIGN) file. All other digital signatures are included within their corresponding resource metadata records in the CATALOG.XML.

An S-100 Exchange Set can be optionally defined alongside S-57 datasets with their own ENC_ROOT and INFO root folders as required by the S-57 ENC Product Specification and (optionally) S-63. In this case there are three top level folders: ENC_ROOT and INFO for S-57, and S100_ROOT with two separate Catalogues covering their respective content (CATALOG.031 and CATALOG.XML). Figure 17-4 below shows one of these use cases with S-57 and multiple S-100 products included.

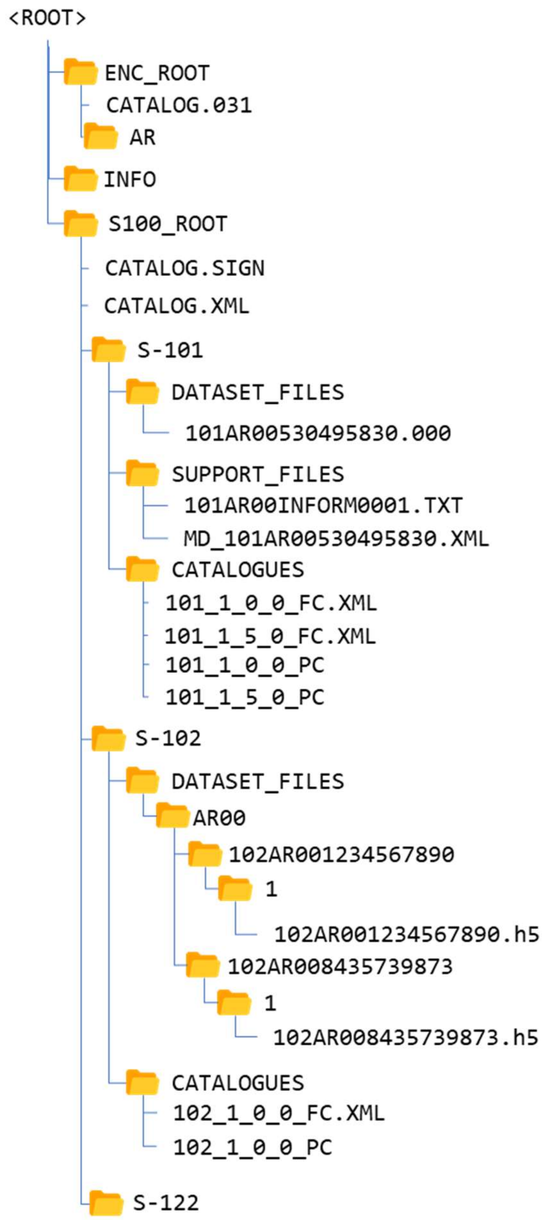


Figure 17-4 – Combined S-100 and S-57 Exchange Sets folder structure

In addition to the folder structure, it is important to align the Exchange Set creation workflow with the data integrity and security provisions outlined in S-100 Part 15. These provisions cover digital signing, compression, and encryption of Exchange Set resources. All resources within an S-100 Exchange Set must be digitally signed and their signatures included in the Exchange Set Catalogue. Data compression and encryption are optional operations.

Exchange Set creation, therefore, consists of:

1. The creation of a suitable Exchange Set folder structure.
2. The arrangement of all resources in their designated folders.
3. Optional compression and encryption of any resources which require it.
4. Creation of digital signatures for all resources.
5. Construction of an Exchange Set Catalogue which records the structure created.

S-100 Part 15 defines the requirements and process for creation and verification of digital signature values and production of compressed/encrypted datasets.

17-4.3 Storage and Management of External Resources

S-100 datasets may refer to a number of externally referenced, supporting resources for content. This content may be textual or graphical and encoded in any of a number of formats (defined by the S100_SupportFileFormat enumeration in the Exchange Catalogue Schema). Datasets hold a reference to the external resource as an attribute value. This value may be updated as any other attribute and updates the reference to the external resource. External resources can support either datasets or Catalogues or can be standalone entities in the Exchange Set

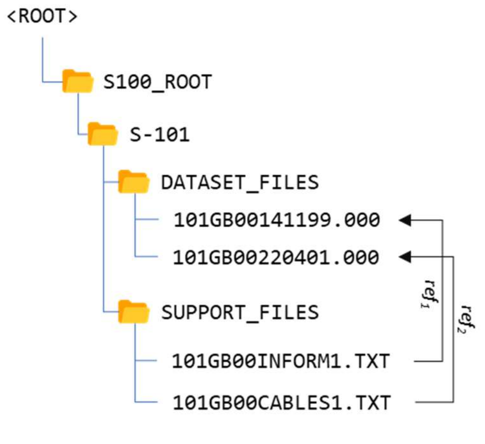
The S-100 Exchange Catalogue provides:

1. A normative definition of the location of each supporting resource. Where these are physical files this is a physical location within the <*S-100 Product*>/SUPPORT_FILES subdirectory in the Exchange Catalogue file structure.
2. For each unique reference to an external resource encoded in a dataset or a supporting resource required by a Catalogue, the Exchange Catalogue provides a unique map (by reference) in the metadata entry for the resource to the dataset or Catalogue metadata entry for which it is required.

For example:

Dataset	Supporting Resource URI
101GB00141199.000	file://101GB00INFORM1.TXT
101GB00220401.000	file://101GB00CABLES1.TXT

Dataset Attribute Values



Exchange Set entries and references

Figure 17-5 – Exchange Set supporting resources (example)

All content relating to such external resources should be validated and must be consistent with the dataset content to form a valid S-100 Exchange Set.

Datasets refer to external resources using S-100 attributes with a URI primitive type. Such references must use S-100 URI form and must be uniquely resolvable by the implementing system without requiring any supplementary information within the Exchange Catalogue metadata entries.

Examples of such URI definitions are:

Table 17-1 – URI references (examples)

URI type	Example URI
File reference	`file::101GB00400797.TXT`
MRN (S-100 Digital Signature Value)	`urn:mrn:iho:s100:dsig:dsa:MEQCIDDzwjK4ksBsMx-AADc5eGQ9uI9Qi8oDx0lVdavMshZnAiBKx_m4KPS3Kk8zYJx-nzeJzhs_H_VHWpVkdtExAqJ-0Q==`

Full specification of file URIs and MRNs supported by S-100 are contained in S-100 Part 1, clause 1-4.6. Use of different types of URIs by ECDIS implementations may be restricted in ECDIS implementations by S-98 Annex C.

As long as the mapping from the external resource metadata to the dataset metadata is unique it is valid, so multiple datasets are able to “share” common external resources within an Exchange Catalogue without ambiguity. To provide unambiguous file URIs from external resources to datasets all base dataset filenames must be unique

Dataset naming shall follow a standard pattern to give implementers the assurance of unique base dataset filenames for incoming datasets.
XXXYYYYØØØØØØØØØ.[EXT]

- XXX is the product code (for example, 123 is for Maritime Radio Services; 101 for ENC)
- YYYY is the producer code according to the Producer Code Register
- ØØØØ is an arbitrary length unique code in alphanumeric characters including any differentiating characters as required. The code shall be unique for the data producer (that is, different data producers may use the same code) and not re-used.
- [EXT] is the file encoding specific file extension

Supporting resources shall follow the same naming convention, except for the ISOMetadataFile which must use the structure MD_<data file base name>.XML. To further assist implementers, data producers shall ensure that the content in the latest revision of supporting resources is specific to the unqiue code used. Differing content in supporting resources shall be assigned different unique codes across an individual Data Producer’s entire content.

A supporting resource cannot be shared across Product Specifications.

Use of the file name in a file URI allows an exchange set producer to maintain a single, up-to-date version of any supporting file resource without necessitating dataset updates when the content of the resource changes. If a data producer wishes to ensure a dataset update is produced whenever supporting resource content changes then use of either digital signature or checksum URIs in the dataset shall be used .

17-4.3.1 Supported resources / multiple references guidance

One single support file can be referenced by feature attributes in multiple datasets. This creates some complexity in a scenario where the support file content is updated, and the changes do not apply to all the datasets currently referencing the support file.

If applicable, in the situation where one support file is referenced by multiple datasets and the support file content changes and initiates a new edition of the support file, all the datasets referencing the support file will adhere to the new edition. If the support file change is not applicable to all datasets referencing it, a new support file must be created for the new changes, and the dataset references to the old support file must be deleted and references to the new support file added. For the datasets not applicable to the change, the old support file and reference will still be valid.

For further detailed explanation refer to S-98.

17-4.3.2 ISOMetadataFile guidance

The S-100 Exchange Set model provides a mechanism for including ISO compliant metadata records for each dataset in an Exchange Set. These optional supporting resources can be included and referenced using the individual ISOMetadataFile records. They are not intended to be used on ECDIS, but may be optionally included to support wider interoperability with other user communities or to fulfil ISO metadata requirements where needed.

17-4.4 S-100 Exchange Set Catalogue

The S100 Exchange Set Catalogue is an XML document instance, which provides the metadata information needed to discover and use the resources contained in the S-100 Exchange Set. It must be named CATALOG.XML. This mandatory, central component of S-100 Exchange Sets consists of several components which capture suitable metadata records for each resource type. These components cover metadata for the Exchange Set Catalogue, dataset discovery, support file discovery, any references to ISO 19115-1/2/3 dataset metadata, and additional Catalogues as depicted in Figure 17-4 above.

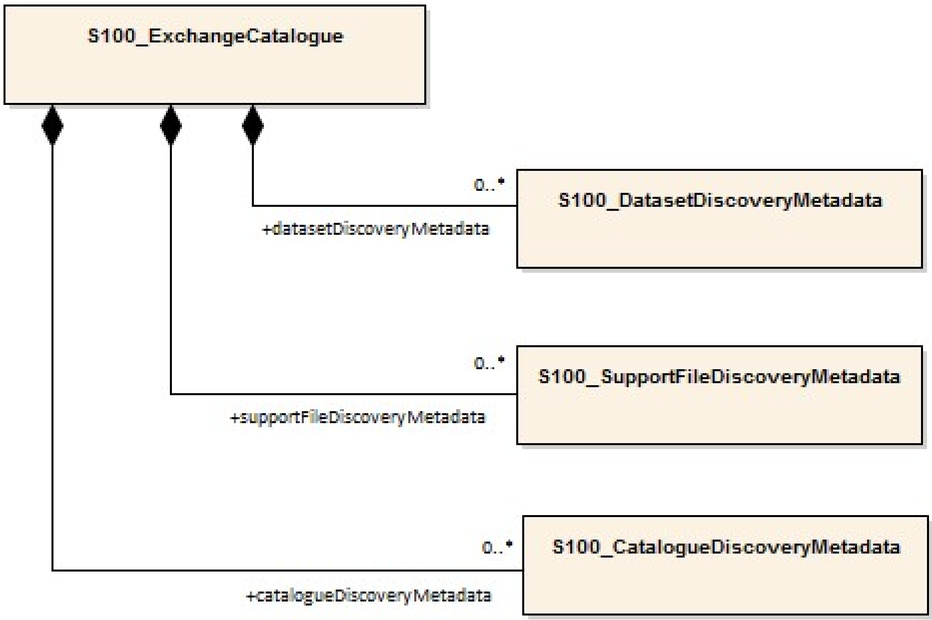


Figure 17-6 – S-100 Exchange Set Catalogue

The discovery metadata subsections have attributes which enable important information about the datasets and accompanying support files to be examined without the need to process the data, for example encryption/compression flags. Similarly, other Catalogues can be included in the S-100 Exchange Set, in support of the datasets, such as feature, portrayal, coordinate reference systems, codelists etc. In addition, the S100 Exchange Set Catalogue provides mechanisms for managing the lifecycle of records, support resources and catalogues. For example, the S100_SupportFileRevisionStatus and S100_Purpose enumerations support a revision control mechanism not only for delivering new versions and revisions, but also for cancelling such resources. This provides the ability to cancel records, support resources and catalogues using the S100 Exchange Set Catalogue records, rather than publishing incremental versions of the actual resources.

More detailed information about the various elements of the Catalogue is shown in Figure 17-7 below and in the textual description in the Tables at clause 17-4.5.

17-4.4.1 New Editions, revisions, updates and cancellations

This section defines the sequencing of datasets for New Editions and, where a particular S-100 encoding supports incremental updates, updates and re-issues. In order to ensure that feature type updates are incorporated into an end user system in the correct sequence without any omission, a number of parameters encoded in the data and metadata are used in the following way:

Edition number When a dataset is initially created (Base dataset), the Edition number 1 is assigned to it. The Edition number is increased by 1 at each New Edition.

Update number Update number 0 is assigned to a new dataset and a New Edition. The first update dataset file associated with this new dataset must have update number 1. The update number must be increased by one for each subsequent update, until a New Edition is released.

A re-issue of a dataset must have the update number of the last update applied to the dataset, and use the same Edition number.

Issue date Date up to which the Data Producer has incorporated all applicable changes. The issue date must be greater than the previous issue date of the dataset.

In addition to fileless dataset cancellation using fields in the Catalogue metadata file a dataset may be cancelled by the Data Producer by the issuing of a cancellation update. In order to cancel a dataset, an update dataset file is created for which the Edition number must be set to 0. This method is only used to cancel a Base dataset file. Where a dataset is cancelled and its name is reused at a later date, the issue date must be greater than the issue date of the cancelled dataset. When the dataset is cancelled it must be removed from the system.

An Exchange Set may contain Base dataset files and update dataset files for the same datasets. Under these circumstances the update dataset files must follow on in the correct sequential order from the last update applied to the Base dataset file.

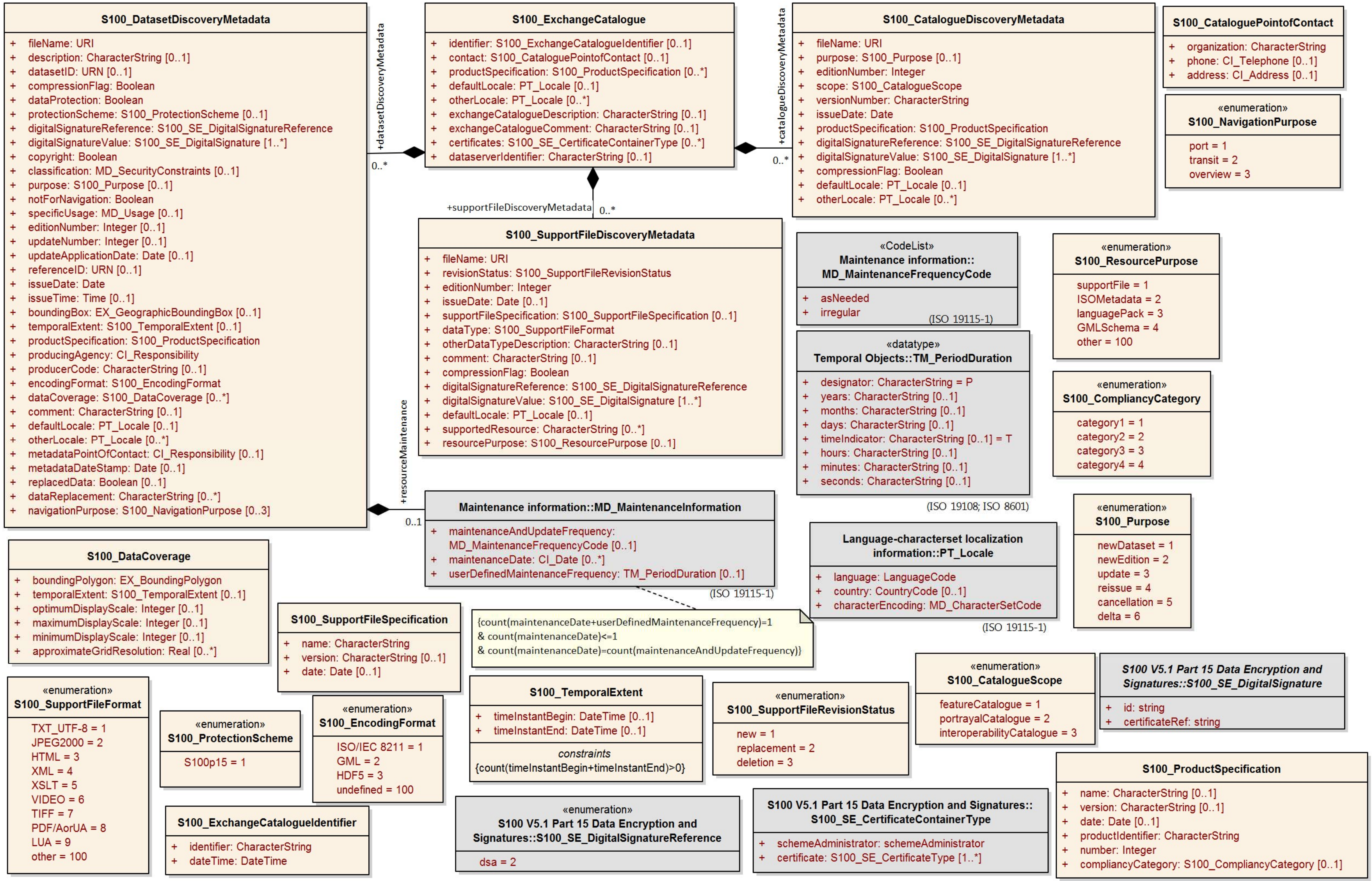


Figure 17-7 – S-100 Exchange Set Catalogue – class details

17-4.5 Elements of the Exchange Set Catalogue

The tables in this section provide a detailed textual description of the encoding of the S-100 Exchange Set Catalogue. The design follows a number of key design principles that have been consistently applied throughout the development process.

One of these principles drives the choice of multiplicity value assignments. At the S-100 framework level, the majority of S-100 Exchange Set Catalogue elements are intended to be optional and therefore have their lower multiplicity bound set to 0. Only those elements that are considered absolutely necessary across all S-100 data products have their lower multiplicity bound set to 1, effectively making them mandatory for all data products. Overall, the resulting multiplicity values at the S-100 Framework level are considered to be a starting point for S-100 Product Specification developers and can be overridden at the individual data product level if necessary.

Another principle was to retain some of the existing element names for historical reasons. For example, the naming of the NotForNavigation element could be improved, but this element was retained from previous versions for backwards compatibility reasons. Similarly, the terms file and resource are used interchangeably in the model and for historical reasons.

S100_ExchangeCatalogue

Each Exchange Set has a single S100_ExchangeCatalogue which contains meta information for the data and support files in the Exchange Set.

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_ExchangeCatalogue	An Exchange Catalogue contains the discovery metadata about the exchange datasets and support files	-	-	-
Attribute	identifier	Uniquely identifies this Exchange Catalogue	0..1	S100_ExchangeCatalogueIdentifier	
Attribute	contact	Details about the issuer of this Exchange Catalogue	0..1	S100_CataloguePointOfContact	
Attribute	productSpecification	Details about the Product Specifications used for the datasets contained in the Exchange Catalogue	0..*	S100_ProductSpecification	
Attribute	defaultLocale	Default language and character set used for all metadata records in this Exchange Catalogue	0..1	PT_Locale	Default is English and UTF-8
Attribute	otherLocale	Other languages and character sets used for the localized metadata records in this Exchange Catalogue	0..*	PT_Locale	Required if any localized entries are present in the Exchange Catalogue
Attribute	exchangeCatalogueDescription	Description of what the Exchange Catalogue contains	0..1	CharacterString	

Attribute	exchangeCatalogueComment	Any additional Information	0..1	CharacterString	
Attribute	certificates	Signed public key certificates referred to by digital signatures in the Exchange Set	0..*	S100_SE_CertificateContainer Type	Content defined in S-100 Part 15. All certificates used, except the SA root certificate (installed separately by the implementing system) shall be included
Attribute	dataServerIdentifier	Identifies the data server for the permit	0..1	CharacterString	
Role	datasetDiscoveryMetadata	Exchange Catalogues may include or reference discovery metadata for the datasets in the Exchange Set	0..*	Aggregation S100_DatasetDiscoveryMetad ata	
Role	catalogueDiscoveryMetadata	Metadata for Catalogue	0..*	Aggregation S100_CatalogueDiscoveryMet adata	Metadata for the Feature, Portrayal and Interoperability Catalogues, if any
Role	supportFileDiscoveryMetadata	Exchange Catalogues may include or reference discovery metadata for the support files in the Exchange Set	0..*	Aggregation S100_SupportFileDiscoveryMe tadata	

S100_ExchangeCatalogueIdentifier

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_ExchangeCatalogueIdentifier	An identifier for an Exchange Catalogue .	-	-	The concatenation of identifier and dateTime form the unique name
Attribute	identifier	Uniquely identifies this Exchange Catalogue	1	CharacterString	<S100XC:identifier>US_101_20200101 _120101_01</S100XC:identifier>
Attribute	dateTime	Creation date and time of the Exchange Catalogue, including time zone	1	DateTime	Format: yyyy-mm-ddThh:mm:ssZ

S100_CataloguePointofContact

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_CataloguePointOfContact	Contact details of the issuer of this Exchange Catalogue	-	-	-
Attribute	organization	The organization distributing this Exchange Catalogue	1	CharacterString	This could be an individual producer, value added reseller, etc
Attribute	phone	The phone number of the organization	0..1	CI_Telephone	

Attribute	address	The address of the organization	0..1	CI_Address	

S100_DatasetDiscoveryMetadata

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_DatasetDiscoveryMetadata	Metadata about the individual datasets in the Exchange Catalogue	-	-	-
Attribute	fileName	Dataset file name	1	URI	See Part 1, clause 1-4.6
Attribute	description	Short description giving the area or location covered by the dataset	0..1	CharacterString	For example, a harbour or port name, between two named locations etc
Attribute	datasetID	Dataset ID expressed as a Marine Resource Name	0..1	URN	The URN must be an MRN
Attribute	compressionFlag	Indicates if the resource is compressed	1	Boolean	*True* indicates a compressed dataset resource *False* indicates an uncompressed dataset resource
Attribute	dataProtection	Indicates if the data is encrypted	1	Boolean	*True* indicates an encrypted dataset resource *False* indicates an unencrypted dataset resource
Attribute	protectionScheme	Specification of method used for data protection	0..1	S100_ProtectionScheme	
Attribute	digitalSignatureReference	Specifies the algorithm used to compute digitalSignatureValue	1	S100_SE_DigitalSignatureReference (see Part 15)	
Attribute	digitalSignatureValue	Value derived from the digital signature	1..*	S100_SE_DigitalSignature (see Part 15)	The value resulting from application of digitalSignatureReference Implemented as the digital signature format specified in Part 15
Attribute	copyright	Indicates if the dataset is copyrighted	1	Boolean	*True* indicates the resource is copyrighted *False* Indicates the resource is not copyrighted

Role Name	Name	Description	Mult	Type	Remarks
Attribute	classification	Indicates the security classification of the dataset	0..1	Class MD_SecurityConstraints>MD_ClassificationCode (codelist)	1. unclassified 2. restricted 3. confidential 4. secret 5. top secret 6. sensitive but unclassified 7. for official use only 8. protected 9. limited distribution
Attribute	purpose	The purpose for which the dataset has been issued	0..1	S100_Purpose	
Attribute	notForNavigation	Indicates the dataset is not intended to be used for navigation	1	Boolean	*True* indicates the dataset is not intended to be used for navigation *False* indicates the dataset is intended to be used for navigation
Attribute	specificUsage	The use for which the dataset is intended	0..1	MD_USAGE>specificUsage (character string)	
Attribute	editionNumber	The Edition number of the dataset	0..1	Integer	When a data set is initially created, the Edition number 1 is assigned to it. The Edition number is increased by 1 at each new Edition. Edition number remains the same for a re-issue
Attribute	updateNumber	Update number assigned to the dataset and increased by one for each subsequent update	0..1	Integer	Update number 0 is assigned to a new dataset
Attribute	updateApplicationDate	This date is only used for the base cell files (that is new data set, re-issue and new edition), not update cell files. All updates dated on or before this date must have been applied by the producer	0..1	Date	
Attribute	referenceID	Reference back to the datasetID	0..1	URN	Update metadata refers to the datasetID of the dataset metadata. This is used if and only if the dataset is an update The URN must be an MRN
Attribute	issueDate	Date on which the data was made available by the Data Producer	1	Date	
Attribute	issueTime	Time of day at which the data was made available by the Data Producer	0..1	Time	The S-100 datatype Time
Attribute	boundingBox	The extent of the dataset limits	0..1	EX_GeographicBoundingBox	-

Role Name	Name	Description	Mult	Type	Remarks
Attribute	temporalExtent	Specification of the temporal extent of the dataset	0..1	S100_TemporalExtent	The temporal extent is encoded as the date/time of the earliest and latest data records (in coverage datasets) or date/time ranges (in vector datasets) If there is more than one feature in a dataset, the earliest and latest time values of records in all features are used, which means the earliest and latest values may be from different features If date/time information for a feature is not encoded in the dataset, it is treated for the purposes of this attribute as extending indefinitely in the appropriate direction on the time axis, limited by the issue date/time or the cancellation or supersession of the dataset This attribute is encoded if and only if at least one of the start and end of the temporal extent is known
Attribute	productSpecification	The Product Specification used to create this dataset	1	S100_ProductSpecification	
Attribute	producingAgency	Agency responsible for producing the data	1	CI_Responsibility>CI_Organisation	See Table 17-3
Attribute	producerCode	The official IHO Producer Code from S-62	0..1	CharacterString	
Attribute	encodingFormat	The encoding format of the dataset	1	S100_EncodingFormat	
Attribute	dataCoverage	Provides information about data coverages within the dataset	0..*	S100_DataCoverage	
Attribute	comment	Any additional information	0..1	CharacterString	
Attribute	defaultLocale	Default language and character set used in the dataset	0..1	PT_Locale	In absence of defaultLocale the language is English, UTF-8
Attribute	otherLocale	Other languages and character sets used in the dataset	0..*	PT_Locale	
Attribute	metadataPointOfContact	Point of contact for metadata	0..1	CI_Responsibility>CI_Individual or CI_Responsibility>CI_Organisation	Only if metadataPointOfContact is different to producingAgency
Attribute	metadataDateStamp	Date stamp for metadata	0..1	Date	May or may not be the issue date
Attribute	replacedData	Indicates if a cancelled dataset is replaced by another data file(s)	0..1	Boolean	See Note

Role Name	Name	Description	Mult	Type	Remarks
Attribute	dataReplacement	Datasetname	0..*	CharacterString	A dataset may be replaced by 1 or more datasets See Note
Attribute	navigationPurpose	Classification of intended navigation purpose (for Catalogue indexing purposes)	0..3	S100_NavigationPurpose	If Product Specification is intended for creation of navigational products this attribute should be mandatory
Role	resourceMaintenance	Information about the frequency of resource updates, and the scope of those updates	0..1	MD_MaintenanceInformation	S-100 restricts the multiplicity to 0..1 and adds specific restrictions on the ISO 19115 structure and content. See clause MD_MaintenanceInformation later in this Part Format: PnYnMnDTnHnMnS (XML built-in type for ISO 8601 *duration*). See clause 17-4.9

NOTE: replacedData and dataReplacement: The intended use of the attributes replacedData and dataReplacement could be, for example, to provide a mechanism for service providers to build automation when providing replacement data sets to customers within existing subscription periods.

S100_NavigationPurpose

Item	Name	Description	Code	Remarks
Enumeration	S100_NavigationPurpose	The navigational purpose of the dataset	-	
Value	port	For port and near shore operations	1	-
Value	transit	For coast and planning purposes	2	-
Value	overview	For ocean crossing and planning purposes	3	-

S100_DataCoverage

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_DataCoverage	A spatial extent where data is provided; and the display scale information for the provided data	-	-	This field is used by user systems as part of the data loading and unloading algorithms and it is strongly encouraged that Product Specifications mandate the use of one or more of the displayScale provided as part of S100_DataCoverage
Attribute	boundingPolygon	A polygon which defines the actual data limit	1..1	EX_BoundingPolygon	

Attribute	temporalExtent	Specification of the temporal extent of the coverage	0..1	S100_TemporalExtent	The remarks for *temporalExtent* in the dataset discovery block (S100_DatasetDiscoveryMetadata) apply, except that their scope is the individual coverage and not the dataset as a whole
Attribute	optimumDisplayScale	The scale with which the data is optimally displayed	0..1	Integer	Example: A scale of 1:25000 is encoded as 25000
Attribute	maximumDisplayScale	The maximum scale with which the data is displayed	0..1	Integer	
Attribute	minimumDisplayScale	The minimum scale with which the data is displayed	0..1	Integer	
Attribute	approximateGridResolution	The resolution of gridded or georeferenced data (in metres)	0..*	Real	A single value may be provided when all axes have a common resolution For multiple value provision, use axis order as specified in dataset May be approximate for ungeorectified data For example, for 5 metre resolution, the value 5 must be encoded * See note

NOTE: approximateGridResolution: If the grid cell size varies over the extent of the grid, an approximated value based on model parameters or production metadata should be used.

S100_Purpose

Item	Name	Description	Code	Remarks
Enumeration	S100_Purpose	The purpose of the dataset	-	
Value	newDataset	Brand new dataset	1	No data has previously been produced for this area
Value	newEdition	New edition of the dataset or Catalogue	2	Includes new information which has not been previously distributed by updates
Value	update	Dataset update	3	Changing some information in an existing dataset
Value	reissue	Dataset that has been re-issued	4	Includes all the updates applied to the original dataset up to the date of the re-issue. A re-issue does not contain any new information additional to that previously issued by updates.
Value	cancellation	Dataset or Catalogue that has been cancelled	5	Indicates the dataset or Catalogue should no longer be used and can be deleted

Value	delta	Dataset difference	6	Reserved for future use

S100_TemporalExtent

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_TemporalExtent	Temporal extent	--		At least one of the timeInstantBegin and timeInstantEnd attributes must be populated; if both are known, both must be populated. The absence of either begin or end indicates indefinite validity in the corresponding direction, limited by the issue date/time or the cancellation or supersession of the dataset
Attribute	timeInstantBegin	The instant at which the temporal extent begins	0..1	DateTime	
Attribute	timeInstantEnd	The instant at which the temporal extent ends	0..1	DateTime	

NOTES:

(1) In case of overlap in temporal extent between predecessor and successor datasets, the successor dataset prevails. For example, water level or weather forecast datasets may have a temporal extent of N days or hours, but be replaced by new forecast at N – X.

(2) Precedence and succession can be determined from information in dataset discovery metadata (for example, attributes for dataReplacement, edition and update numbers, issue data and time).

EXAMPLE 1: An S-104 (Water Level Information for Surface Navigation) predictions dataset has the following data for *temporalExtent* encoded in the dataset discovery block in the Exchange Catalogue:

```
<temporalExtent>
  <timeInstantBegin>2021-07-03T06:00:00Z</timeInstantBegin>
  <timeInstantEnd>2021-07-10T18:00:00Z</timeInstantEnd>
</temporalExtent>
```

indicating that the temporal extent of the predictions in the dataset is the period beginning at exactly 6 a.m. on 3 July 2021 (UTC) and ending at exactly 6 p.m. on 10 July 2021 (UTC).

EXAMPLE 2: The successor dataset to Example 1 has the following data for *temporalExtent*:

```
<temporalExtent>
  <timeInstantBegin>2021-07-03T12:00:00Z</timeInstantBegin>
```

```
    <timeInstantEnd>2021-07-10T24:00:00Z</timeInstantEnd>
  </temporalExtent>
```

indicating that the temporal extent of the predictions in the dataset is the period beginning at exactly noon on 3 July 2021 (UTC) and ending at exactly midnight at the end of 10 July 2021 (UTC). Since this temporal extent overlaps the temporal extent of Example 1 from noon UTC on 3 July 2021, it supersedes the dataset in Example 1 at and after noon UTC on 3 July 2021.

S100_EncodingFormat

Item	Name	Description	Code	Remarks
Enumeration	S100_DataFormat	The encoding format	-	-
Value	ISO/IEC 8211	The ISO 8211 data format as defined in Part 10a	1	-
Value	GML	The GML data format as defined in Part 10b	2	-
Value	HDF5	The HDF5 data format as defined in Part 10c	3	-
Value	undefined	The encoding is defined in the Product Specification	100	Use of Product Specification specific encoding means the data product and Product Specification is not intended for an IHO S-100 compliant system

S100_ProductSpecification

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_ProductSpecification	The Product Specification contains the information needed to build the specified product	-	-	-
Attribute	name	The name of the Product Specification used to create the datasets	0..1	CharacterString	The name in the GI Registry should be used for this field. For example, "Electronic Navigational Chart"
Attribute	version	The version number of the Product Specification	0..1	CharacterString	TR 2/2007 specifies versioning of Product Specifications.
Attribute	date	The version date of the Product Specification	0..1	Date	
Attribute	productIdentifier	Machine readable unique identifier of a product type	1	CharacterString (Restricted to Product ID values from the IHO Product Specification Register, in the IHO Geospatial Information Registry)	For example, "S-101"

Attribute	number	The number used to lookup the product in the Product Specification Register of the IHO GI registry	1	Integer	For IHO Product Specifications these should be taken from the IHO Product Specification Register in the IHO Geospatial Information (GI) Registry
Attribute	compliancyCategory	The level of compliance of the Product Specification to S-100	0..1	S100_CompliancyCategory	See Part 4a, clause 4a-5.5

S100_CompliancyCategory

Item	Name	Description	Code	Remarks
Enumeration	S100_CompliancyCategory		-	-
Value	category1	IHO S-100 object model compliant	1	
Value	category2	IHO S-100 compliant with non-standard encoding	2	
Value	category3	IHO S-100 compliant with standard encoding	3	
Value	category4	IHO S-100 and IMO harmonized display compliant	4	

S100_ProtectionScheme

Item	Name	Description	Code	Remarks
Enumeration	S100_ProtectionScheme	Data protection schemes	-	-
Value	S100p15	IHO S-100 Part 15	1	See Part 15

S100_SupportFileDiscoveryMetadata

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_SupportFileDiscoveryMetadata	Metadata about the individual support files in the Exchange Catalogue	-	-	-
Attribute	fileName	Name of the support file	1	URI	See Part1, clause 1-4.6
Attribute	revisionStatus	The purpose for which the support file has been issued	1	S100_SupportFileRevisionStatus	For example new, replacement, etc

Role Name	Name	Description	Mult	Type	Remarks
Attribute	editionNumber	The Edition number of the support file	1	Integer	When a data set is initially created, the Edition number 1 is assigned to it. The Edition number is increased by 1 at each new Edition. Edition number remains the same for a re-issue
Attribute	issueDate	Date on which the data was made available by the Data Producer	0..1	Date	
Attribute	supportFileSpecification	The specification used to create this file	0..1	S100_SupportFileSpecification	
Attribute	dataType	The format of the support file	1	S100_SupportFileFormat	
Attribute	otherDataTypeDescription	Support file format other than those listed	0..1	CharacterString	
Attribute	comment	Optional comment	0..1	CharacterString	
Attribute	compressionFlag	Indicates if the resource is compressed	1	Boolean	*True* indicates a compressed resource *False* indicates an uncompressed resource
Attribute	digitalSignatureReference	Specifies the algorithm used to compute digitalSignatureValue	1	S100_SE_DigitalSignatureReference (see Part 15)	
Attribute	digitalSignatureValue	Value derived from the digital signature	1..*	S100_SE_DigitalSignature (see Part 15)	The value resulting from application of digitalSignatureReference Implemented as the digital signature format specified in Part 15
Attribute	defaultLocale	Default language and character set used in the support file	0..1	PT_Locale	In absence of defaultLocale the language is English in UTF-8 A support file is expected to use only one as locale. Additional support files can be created for other locales
Attribute	supportedResource	Identifier of the resource supported by this support file	0..*	CharacterString	Conventions for identifiers are detailed in S-100 Part 15. S-100 allows file URI, digital signature or cryptographic hash checksums to be used
Attribute	resourcePurpose	The purpose of the supporting resource	0..1	S100_ResourcePurpose	Identifies how the supporting resource is used

S100_SupportFileFormat

Item	Name	Description	Code	Remarks
Enumeration	S100_SupportFileFormat	The format used for the support file	-	-
Value	TXT_UTF-8	UTF-8 text excluding control codes	1	-
Value	JPEG2000	JPEG2000 format	2	ISO 15444
Value	HTML	Hypertext Markup Language	3	
Value	XML	Extensible Markup Language	4	
Value	XSLT	Extensible Stylesheet Language Transformations	5	
Value	VIDEO	Representation of moving images in unspecified format	6	
Value	TIFF	Tagged Image File Format	7	
Value	PDF/AorUA	Portable Document Format	8	ISO 19005, ISO 32000 Product Specification developers should take careful consideration in using PDF as a support file format. It is recommended that PDF never be used in products that will be used on a navigation system as it may impair night vision Must be PDF/A or UA
Value	LUA	Lua programming language	9	
Value	other	Other format	100	

S100_SupportFileRevisionStatus

Item	Name	Description	Code	Remarks
Enumeration	S100_SupportFileRevisionStatus	The reason for inclusion of the support file in this Exchange Set	-	-
Value	new	A file which is new	1	Signifies a new file
Value	replacement	A file which replaces an existing file	2	Signifies a replacement for a file of the same name
Value	deletion	Deletes an existing file	3	Signifies deletion of a file of that name

S100_SupportFileSpecification

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_SupportFileSpecification	The standard or specification to which a support file conforms	-	-	-
Attribute	name	The name of the specification used to create the support file	1	CharacterString	
Attribute	version	The version number of the specification	0..1	CharacterString	
Attribute	date	The version date of the specification	0..1	Date	

S100_ResourcePurpose

Item	Name	Description	Code	Remarks
Enumeration	S100_ResourcePurpose	Defines the purpose of the supporting resource	-	-
Value	supportFile	A support file	1	
Value	ISOMetadata	Dataset metadata in ISO format	2	
Value	languagePack	A Language pack	3	
Value	GMLSchema	GML Application Schema	4	
Value	other	A type of resource not otherwise described	100	

S100_CatalogueDiscoveryMetadata

Role Name	Name	Description	Mult	Type	Remarks
Class	S100_CatalogueDiscoveryMetadata	Class for S-100 Catalogue metadata	-	-	-
Attribute	fileName	The name for the Catalogue	1	URI	See Part1, clause 1-4.6
Attribute	purpose	The purpose for which the Catalogue has been issued	0..1	S100_Purpose (codelist)	The values must be one of the following: *2* new edition *5* cancellation Default is new edition

Attribute	editionNumber	The Edition number of the Catalogue	1	Integer	Initially set to 1 for a given productSpecification.number Increased by 1 for each subsequent newEdition Uniquely identifies the version of the Catalogue
Attribute	scope	Subject domain of the Catalogue	1	S100_CatalogueScope	
Attribute	versionNumber	The version identifier of the Catalogue	1	CharacterString	Human readable version identifier
Attribute	issueDate	The issue date of the Catalogue	1	Date	
Attribute	productSpecification	The Product Specification used to create this file	1	S100_ProductSpecification	
Attribute	digitalSignatureReference	Specifies the algorithm used to compute digitalSignatureValue	1	S100_SE_DigitalSignatureReference (see Part 15)	
Attribute	digitalSignatureValue	Value derived from the digital signature	1..*	S100_SE_DigitalSignature (see Part 15)	The value resulting from application of digitalSignatureReference Implemented as the digital signature format specified in Part 15
Attribute	compressionFlag	Indicates if the resource is compressed	1	Boolean	*True* indicates a compressed resource *False* indicates an uncompressed resource
Attribute	defaultLocale	Default language and character set used in the Catalogue	0..1	PT_Locale	In absence of defaultLocale the language is English in UTF-8
Attribute	otherLocale	Other languages and character sets used in the Catalogue	0..*	PT_Locale	

S100_CatalogueScope

Item	**Name**	**Description**	**Code**	**Remarks**
Enumeration	S100_CatalogueScope	The scope of the Catalogue	-	-
Value	featureCatalogue	S-100 Feature Catalogue	1	
Value	portrayalCatalogue	S-100 Portrayal Catalogue	2	
Value	interoperabilityCatalogue	S-100 Interoperability Catalogue	3	

MD_MaintenanceInformation

Role Name	Name	Description	Mult	Type	Remarks
Class	MD_MaintenanceInformation	Information about the scope and frequency of updating	-	-	S-100 restricts the ISO 19115-class to: • prohibit maintenanceScope, maintenanceNote, and contact attributes; • define restrictions on maintenanceAndUpdate Frequency, maintenanceDate, and userDefinedMaintenance Frequency attributes
Attribute	maintenanceAndUpdateFrequency	Frequency with which changes and additions are made to the resource after the initial resource is completed	0..1	MD_MaintenanceFrequencyCode (codelist)	Must be populated if userDefined MaintenanceFrequency is not present, otherwise optional. See Table MD_MaintenanceFrequencyCode in this Part for values allowed in S-100 metadata
Attribute	maintenanceDate	Date information associated with maintenance of the resource	0..1	CI_Date	Exactly one of maintenanceDate and userDefinedMaintenanceFrequency must be populated Allowed value for dateType: nextUpdate
Attribute	userDefinedMaintenanceFrequency	Maintenance period other than those defined	0..1	TM_PeriodDuration	Exactly one of maintenanceDate and userDefinedMaintenanceFrequency must be populated Only positive durations allowed

See clause 17-4.9 for more information about encoding maintenance information.

MD_MaintenanceFrequencyCode

S-100 uses a subset of the values allowed in ISO 19115-1.

Item	Name	Description	Code	Remarks
Enumeration	MD_MaintenanceFrequencyCode	Frequency with which modifications and deletions are made to the data after it is first produced	-	S-100 is restricted to only the following values from the ISO 19115-1 codelist. The conditions for the use of a particular value are described in its Remarks
Value	asNeeded	Resource is updated as deemed necessary	1	Use only for datasets which normally use a regular interval for update or supersession, but will have the next update issued at an interval different from the usual Allowed if and only if userDefinedMaintenanceFrequency is not populated

Value	irregular	Resource is updated in intervals that are uneven in duration	2	Use only for datasets which do not use a regular schedule for update or supersession Allowed if and only if userDefinedMaintenanceFrequency is not populated

CI_DateTypeCode

This codelist is documented in the ISO Schemas documentation, available in the S-100 Schemas distribution. It is used in several places in S-100 metadata.

PT_Locale

Role Name	Name	Description	Mult	Type	Remarks
Class	PT_Locale	Description of a locale	-	-	From ISO 19115-1
Attribute	language	Designation of the locale language	1	LanguageCode	ISO 639-2/T 3-letter language codes.
Attribute	country	Designation of the specific country of the locale language	0..1	CountryCode	ISO 3166-2 2-letter country codes
Attribute	characterEncoding	Designation of the character set to be used to encode the textual value of the locale	1	MD_CharacterSetCode	UTF-8 is used in S-100

Table 17-2 – Individuals (restriction of CI_Individual from ISO 19115-1)

Name	Path	Datasets	Other resources
Name of the individual	CI_Individual.name	**C** *(documented if 'positionName' and 'partyIdentifier' not documented)*	**C** *(same as for dataset)*
Position of the individual in an organization	CI_Individual.positionName	**C** *(documented if 'name' and 'partyIdentifier' not documented)*	**C** *(same as for dataset)*
Contact information for the individual	CI_Individual > contactInfo > CI_Contact	**M** (see note 2)	**M** (see note 2)
Identifier for the party	CI_Individual.partyIdentifier	**C** *(documented if 'name' and 'positionName' not documented*	**C** *(same as for dataset)*

Table 17-3 – Organisations (restriction of CI_Organisation from ISO 19115-1)

Name	Path	Datasets	Other resources
Name of the organisation	CI_Organisation.name	**C** *(documented if 'positionName' not documented* – see Note 1*)*	**C** *(same as for dataset)*
Position of an individual in the organisation	CI_Organisation.positionName	**C** *(documented if 'name' not documented* – see Note 1*)*	**C** *(same as for dataset)*
Contact information for the organisation	CI_Organisation.contactInfo > CI_Contact	**M** (see note 2)	**M** (see note 2)
Identifier for the party	CI_Organisation.partyIdentifier	**C** *(documented if 'name' and 'positionName' not documented*	**C** *(same as for dataset)*

NOTE 1 S-100 restricts ISO 19115-1 in that documenting the 'logo' attribute of CI_Organisation is not sufficient to allow omission of both 'name' and 'positionName'.

NOTE 2 At least one of CI_Contact attributes phone / address / onlineResource / contactInstructions must be documented.

17-4.6 Overview of multilingual support in S-100 Exchange Set Catalogue

The S100 Exchange Set Catalogue provides the necessary multilingual support by directly reusing the localization framework present in ISO 19115-1:2014 metadata standard. This effectively adds two localization elements: defaultLocale and otherLocale to various classes within the model. These elements are intended to consistently identify the languages used in both the metadata records and within the geospatial resources, such as datasets, support files and other Catalogues included in an Exchange Set. The defaultLocale element is intended to identify the default language and character set while the otherLocale element is intended to provide the same for any alternatively used localized character strings. Both elements are defined as PT_Locale type defined as illustrated in Figure 17-8 below.

PT_Locale
+ language: LanguageCode
+ country: CountryCode [0...1]
+ characterEncoding: MD_CharacterSetCode

Figure 17-8 – ISO 19115-1:2014 PT_Locale class.

The PT_Locale class as defined in ISO 19115-1:2014 has the following members:

- LanguageCode – required ISO 639-2/T, 3-letter code in lowercase; that is, “fra”
- CountryCode – optional ISO 3166-1 2-letter code in uppercase; that is,“CA” intended to be used when the national language differences can impact the interpretation or processing of localized content
- MD_CharacterSetCode – required MD_CharacterSetCode

NOTE: Since codes for language, country, and character sets are defined as entries in a “codelists catalogue” that is included in the S-100 Schema distribution, the codelist values must be identical to keys in this file.

EXAMPLE: The codelist value for LanguageCode is 'eng'. It is encoded in the XML attribute `codeListValue`.

<lan:LanguageCode codeList="http://www.iho.int/S100/5.0.0/resources/Codelists/cat/codelists.xml#S100_MD_LanguageCode" codeListValue="eng">English</lan:LanguageCode>

For more details and examples, see the documentation and samples provided with the S-100 generic Schemas.

The implementation of the PT_Locale type provides the necessary structure to consistently define and communicate the key language characteristics within metadata or other geospatial resources.

Additionally, the localization framework provides the support for using multiple languages in the metadata records by extending CharacterString simple type with PT_FreeText and LocalisedCharacterString subtypes as illustrated in Figure 17-9 below.

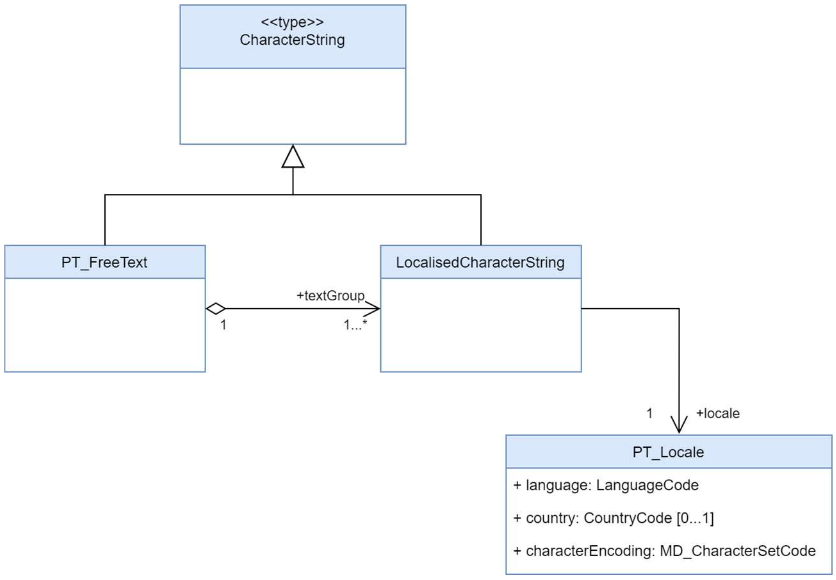


Figure 17-9 – ISO 19115-1:2014 PT_FreeText and LocalisedCharacterString subtypes

This allows any free text metadata record instances expressed in the default metadata language to also be expressed in other languages by aggregating the corresponding localized translations using LocalisedCharacterString and adding a reference to the underlying otherLocale definition. The diagram below shows a pseudo-XML implementation example illustrating how such aggregations should be constructed.

```xml
<S100XC:metadata_record xsi:type="lan:PT_FreeText_PropertyType">
    <gco:CharacterString>text expressed using defaultLocale</gco:CharacterString>
    <lan:PT_FreeText>
        <lan:textGroup>
            <lan:LocalisedCharacterString locale="#referece_to_otherLocale">text expressed using otherLocale</lan:LocalisedCharacterString>
        </lan:textGroup>
    </lan:PT_FreeText>
</S100XC:metadata_record>
```

17-4.7 Encoding of S-100 Exchange Set Catalogue elements in multiple languages

The S100 Exchange Set Catalogue model provides two elements: defaultLocale and otherLocale to define and indicate the languages used for all metadata records within an instance of an Exchange Catalogue. Only one defaultLocale is permitted within the core section of the S100 Exchange Set Catalogue (within S100_ExchangeCatalogue) and it is intended to communicate the default language used for all Catalogue records. Since the expected default language is English and the default character set is UTF-8 the defaultLocale element is optional and can be omitted. In most situations, however, it is prudent to explicitly define defaultLocale to prevent any confusion and more readily support data sharing with other user communities that might not be fully aware of S-100 conventions. This can be achieved as illustrated below.

```xml
<S100XC:defaultLocale>
    <lan:PT_Locale>
        <lan:language>
            <lan:LanguageCode codeList="http://www.iho.int/S100/ ... #S100_MD_LanguageCode" codeListValue="eng">English</lan:LanguageCode>
        </lan:language>
        <lan:characterEncoding>
            <lan:MD_CharacterSetCode codeList="http://www.iho.int/S100/ ... #S100_MD_CharacterSetCode" codeListValue="utf8">UTF-8</lan:MD_CharacterSetCode>
        </lan:characterEncoding>
    </lan:PT_Locale>
</S100XC:defaultLocale>
```

Data producing agencies wishing to provide additional localized translations of any of the Catalogue records can achieve so by first defining otherLocale and then referring to it when required. The first step can be achieved as illustrated below and, similarly to defaultLocale, this only needs to be defined once within the core section of the S100 Exchange Set Catalogue (within S100_ExchangeCatalogue) for each additional language used in a Catalogue instance. This approach is intended to communicate any additional language used for localized Catalogue records.

```
<S100XC:otherLocale>
    <lan:PT_Locale id="locale_fra_ca">
        <lan:language>
            <lan:LanguageCode codeList="http://www.iho.int/S100/ ... #S100_MD_LanguageCode" codeListValue="fra">Français</lan:LanguageCode>
        </lan:language>
        <lan:country>
            <lan:CountryCode codeList="http://www.iho.int/S100/ ... #S100_MD_CountryCode" codeListValue="CA">Canada</lan:CountryCode>
        </lan:country>
        <lan:characterEncoding>
            <lan:MD_CharacterSetCode codeList="http://www.iho.int/S100/ ... #S100_MD_CharacterSetCode" codeListValue="utf8">UTF-8</lan:MD_CharacterSetCode>
        </lan:characterEncoding>
    </lan:PT_Locale>
</S100XC:otherLocale>
```

Of note is the id attribute of PT_Locale. When used in otherLocale definition, it needs to be a unique, ideally descriptive identification of a specific language which can be used as a reference by localized records. With the otherLocale element defined, any free text instances captured using the default language can also provide corresponding localized translations using PT_FreeText and LocalisedCharacterString subtypes as illustrated below.

```
<S100XC:specificUsage>
    <mri:MD_Usage>
        <mri:specificUsage xsi:type="lan:PT_FreeText_PropertyType">
            <gco:CharacterString>Coastal Navigation</gco:CharacterString>
            <lan:PT_FreeText>
                <lan:textGroup>
                    <lan:LocalisedCharacterString locale="#locale_fra_ca">Navigation côtière</lan:LocalisedCharacterString>
                </lan:textGroup>
            </lan:PT_FreeText>
        </mri:specificUsage>
    </mri:MD_Usage>
</S100XC:specificUsage>
```

17-4.8 Indicating languages used inside geospatial resources described in S-100 Exchange Set Catalogue

Data producing agencies using multiple languages in their products or other resources, who wish to explicitly indicate the languages used can use the same localization framework. In contrast to the metadata records, where language definitions are applicable to all records in an Exchange Catalogue instance, the default and other language definitions are individual resource specific. This is accomplished by defining default and/or other languages in the same way as before but placing them inside specific resource records. For example, a data producing agency wishing to communicate that a specific dataset includes features encoded using multiple languages can add the defaultLocale and otherLocale definitions inside the corresponding dataset discovery metadata record. At the resource level, both of these elements are optional and English UTF-8 encoding is considered to be the default therefore there is generally no need to capture this fact explicitly.

S-100 support file resources are a special case, as the textual information inside them is intended to be in a single language. As with all other resources, English UTF-8 encoding is the default therefore there is no need to capture this fact explicitly. It would be prudent, however, to define support file specific defaultLocale when the language used for the content is other than English. Both the S-100 Exchange Set Catalogue and S-100 Datasets can reference any number of support resources. The diagram below shows a pseudo-XML version with examples of MRN-based identifiers used as references between datasets and support resources. This illustrates the mechanism for using a predefined referencing system to interconnect the independently captured metadata records for datasets and support resources.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<S100_ExchangeCatalogue>
    <datasetDiscoveryMetadata>
        <S100_DatasetDiscoveryMetadata>
            <fileLocation>DATASET_FILES/101GB00141199.000</fileLocation>
            <digitalSignatureValue>
                  302C0214394435532378E3E5038E56DB1054174D4B445C6402143D41CBAD47BBBBA863CC3D8C270F27C532A39151
            </digitalSignatureValue>
            <supportFileDiscoveryMetadataReference>
                  urn:mrn:iho:sha256:6003c71cca084c4c1e285722391bb00f0e47bbd672e316d9842c3b93a6f6e050
            </supportFileDiscoveryMetadataReference>

        </S100_DatasetDiscoveryMetadata>
        <S100_DatasetDiscoveryMetadata>
            <fileLocation>DATASET_FILES/101GB00220401.000</fileLocation>
            <digitalSignatureValue>
                  302C0214145339C46C30382CEED4655F850D1877A873D67502140D5168EBC665C02A3C8A20163CCC60E82FA7A306
            </digitalSignatureValue>

            <supportFileDiscoveryMetadataReference>
                  urn:mrn:iho:sha256:27f0d4fe98d9f19608837e68bea0d1f3ee0c6f8defbcb02a5a9f6ab4d9459d67
            </supportFileDiscoveryMetadataReference>
OR:
            <supportFileDiscoveryMetadataReference>
                  urn:mrn:iho:dsig:302C02147CFBFBD2A731BA7CD6292C06942DF4D4413468A50214191C9373DD3BD87FB7EEE760275F7A422F77BD4C
            </supportFileDiscoveryMetadataReference>


        </S100_DatasetDiscoveryMetadata>
    </datasetDiscoveryMetadata>
    <supportFileDiscoveryMetadata>
        <fileLocation>SUPPORT_FILES/101GB00INFORM1.TXT</fileLocation>
        <digitalSignatureValue>
                  302C021435C50AAC5B7E8D8395EEF4EDFA48420490376FD0021479F0083DD577654E0ABA68C80721B4883EE3EE76
        </digitalSignatureValue>
    </supportFileDiscoveryMetadata>
    <supportFileDiscoveryMetadata>
        <fileLocation>SUPPORT_FILES/101GB00CABLES1.TXT</fileLocation>
        <digitalSignatureValue>
                  302C02147CFBFBD2A731BA7CD6292C06942DF4D4413468A50214191C9373DD3BD87FB7EEE760275F7A422F77BD4C
        </digitalSignatureValue>
    </supportFileDiscoveryMetadata>
</S100_ExchangeCatalogue>
```

The above diagram also illustrates the optional defaultLocale fully omitted for any resources encoded using English UTF-8 thus simplifying the related metadata content. At the same time, data producers wishing to supply support resources in other languages can achieve this by capturing them independently and adding the corresponding metadata records, including defining their defaultLocale, as appropriate. The diagram below shows a pseudo-XML metadata example of a support resource supplied as two individual files one in English and the other in French.

```xml
<supportFileDiscoveryMetadata>
    <fileLocation>SUPPORT_FILES\101CACAB01_ENG.TXT</fileLocation>
    <digitalSignatureValue>642C559A55029FBC85CA86459DB769ABDE7C5E5CE631D32E789EA15E0805EE41</digitalSignatureValue>
 </supportFileDiscoveryMetadata>

<supportFileDiscoveryMetadata>
    <fileLocation>SUPPORT_FILES\101CACAB01_FRA.TXT</fileLocation>
    <digitalSignatureValue>E3B0C44298FC1C149AFBF4C8996FB92427AE41E4649B934CA495991B7852B855</digitalSignatureValue>
    <S100XC:defaultLocale>
        <lan:PT_Locale id="locale_fra_ca">
            <lan:language>
                <lan:LanguageCode codeList="http://www.iho.int/S100/ ... #S100_MD_LanguageCode" codeListValue="fra">Français</lan:LanguageCode>
            </lan:language>
            <lan:country>
                <lan:CountryCode codeList="http://www.iho.int/S100/ ... #S100_MD_CountryCode" codeListValue="CA">Canada</lan:CountryCode>
            </lan:country>
            <lan:characterEncoding>
                <lan:MD_CharacterSetCode codeList="http://www.iho.int/S100/ ... #S100_MD_CharacterSetCode" codeListValue="utf8">UTF-8</lan:MD_CharacterSetCode>
            </lan:characterEncoding>
        </lan:PT_Locale>
    </S100XC:defaultLocale>
</supportFileDiscoveryMetadata>
```

While the localization framework currently allows a high degree of flexibility, data producers are strongly encouraged to apply one consistent multilanguage support approach across their entire S-100 product portfolios to ensure a consistent user experience. The recommended approach is to provide all support resources in any other officially supported language in addition to English.

17-4.9 Encoding of maintenance information

The interval described by *userDefinedMaintenanceFrequency* is with respect to the issue date and time of the dataset described by this dataset discovery metadata block. End-user's and distributor's systems should use this interval for planning any automated operations to obtain the successor dataset, but must allow for delays or variations in the actual availability of successor dataset(s).

The format for *userDefinedMaintenanceFrequency* is given by the XML built-in datatype *duration*, which can be validated by off-the-shelf XML parsers. See "*XML Schema Part 2: Datatypes (2nd edition) - Clause 3.2.6 duration*" (relevant extracts below):

> The lexical representation for **duration** is the ISO 8601 extended format PnYnMnDTnHnMnS, where nY represents the number of years, nM the number of months, nD the number of days, 'T' is the date/time separator, nH the number of hours, nM the number of minutes and nS the number of seconds. The number of seconds can include decimal digits to arbitrary precision.
>
> The values of the Year, Month, Day, Hour and Minutes components are not restricted but allow an arbitrary unsigned integer; that is, an integer that conforms to the pattern [0-9]+.. Similarly, the value of the Seconds component allows an arbitrary unsigned decimal. Following ISO 8601, at least one digit must follow the decimal point if it appears.
>
> Reduced precision and truncated representations of this format are allowed provided they conform to the following:
>
> - If the number of years, months, days, hours, minutes, or seconds in any expression equals zero, the number and its corresponding designator ·may· be omitted. However, at least one number and its designator ·must· be present.
> - The seconds part ·may· have a decimal fraction.
> - The designator 'T' must be absent if and only if all of the time items are absent. The designator 'P' must always be present.

17-4.9.1 Encoding and interpretation rules in S-100 metadata

(1) Restriction to non-negative durations: S-100 restricts the duration type by prohibiting zero or negative values of duration in userDefinedMaintenanceFrequency.

(2) Number of digits: S-100 recommends (but does not require) using 2 digits for the months, days, hours, minutes, components, when they are present. If the seconds component is encoded, two digits are recommended for the number of whole seconds (for example, encode 0.5 seconds as PT00.5S; encode 100 seconds as PT01M40S).

(3) Start and end instants: The start and end instants of the interval calculated by combining userDefinedMaintenanceFrequency with the issue date/time must be interpreted according to Part 3 Clause 3-8. The value must be encoded appropriately; this means that smaller date/time components must not be encoded unless the availability of the successor dataset is known to the corresponding level of precision. Smaller units should be used when the availability is known to the corresponding precision, such as "48 hours" instead of "2 days" when the successor dataset availability is planned to the hour.

(4) Encoding of zero components: Zero components must be encoded if and only if they are significant for indicating the granularity of the start/end instants of the interval.

(5) Variability: A variation of ±X should be allowed for, where X is the component of smallest granularity; if the value of the smallest component is 1, variability is unspecified.

(6) Stability for successive datasets, and exceptions: The value of this attribute will normally be stable over a sequence of predecessor/successor datasets. The alternate encoding using maintenanceDate should be used for known exceptional circumstances affecting the release of a successor, such as an office closure at the end of the intervening period, reverting to normal encoding with userDefinedMaintenanceFrequency when the normal update schedule is restored.

(7) Off-schedule updates: Communication of exceptional, unforeseeable off-schedule issues of data such as emergency hurricane forecasts should be provided for by other means than userDefinedMaintenanceFrequency or maintenanceDate attributes, since they are by definition unforeseeable.

(8) Supersession: If both userDefinedMaintenanceFrequency and maintenanceDate are encoded in the same discovery metadata block, the maintenanceDate supersedes the userDefinedMaintenanceFrequency.

EXAMPLES:

Table 17-4 – Maintenance metadata (examples)

No.	maintenanceAndUpdateFrequency	maintenanceDate	userDefined Maintenance Frequency	Remarks
1	--	--	P3DT10H30M	An interval of 3 days, 10 hours, and 30 minutes. Variability +/-1 minute.
2	--	--	PT6H	An interval of exactly 6 hours, with a variability of +/1 hour.
3	--	--	P30M	An interval of 30 months.
4	--	--	PT30M	An interval of 30 minutes.
5	--	--	P6H P30S P30M10S	Invalid (they contain time components but lack the 'T' designator)
6	--	--	PT30m	Invalid ('m' should be upper-case).
7	--	--	PT12:30 P3DT10H 30M	Invalid (the ':' or space separators are not allowed, only the separators specified by the XML Schema datatypes specification for duration are allowed)
8	--	--	P1M	One month, variability unknown. According to the "Start and end instants" rule, will be interpreted as the same day in the following month, or the nearest preceding day if there is no such date in the following month. If the issue date of the current dataset is 30 August, the successor dataset can be expected to be issued between midnight at the beginning of 30 September and midnight at the end of 30 September.
9	--	--	P1M00D	One month, with a variability of +/- 1 day. With a dataset issued on January 31 2021, the next dataset is expected on February 28, 2021; with a dataset issued on January 31, 2024 means the next dataset is expected February 29, 2024. A 1-day variation before after those dates should be anticipated.
10	--	--	P30D	30 days, variability +/- 1 day. With a dataset issued on January 31, 2021 it means the next dataset is expected on March 2, 2021; with a dataset issued on January 31, 2024 it means the next dataset is expected on March 1, 2024. A 1-day variation should be allowed for in both cases.

11	irregular	cit:CI_Date > cit:dateType=nextUpdate cit:date=2021-10-25	--	On 25 October 2021, at an unspecified time on that date.
12	irregular	cit:CI_Date > cit:dateType=nextUpdate cit:date=2021-10-25T14:00:00Z	--	On 25 October 2021, at 2 pm UTC.
13	asNeeded	cit:CI_Date > cit:dateType=nextUpdate cit:date=2021-10-25T14:00:00Z	--	To encode an exception to a dataset sequence normally on a regular schedule. Next dataset will be available on 25 October 2021, at 2 pm UTC.

XML encoding examples:

EXAMPLE 1: Dataset is updated at an interval of 6 hours:

```
<mri:resourceMaintenance>
  <mmi:MD_MaintenanceInformation>
    <mmi:userDefinedMaintenanceFrequency>
      <gco:TM_PeriodDuration>PT06H</gco:TM_PeriodDuration>
    </mmi:userDefinedMaintenanceFrequency>
  </mmi:MD_MaintenanceInformation>
</mri:resourceMaintenance>
```

EXAMPLE 2: Dataset is normally updated on a regular schedule, but the next update will be on 1 January 2022 at 5 am local time in the time zone with UTC offset -5 hours (for example, 5 am US Eastern Standard Time). The codeList attributes must be populated with the URL of the appropriate codelist, which will be in the ISO or S-100 Schema distribution package.

```
<mri:resourceMaintenance>
  <mmi:MD_MaintenanceInformation>
    <mmi:maintenanceAndUpdateFrequency>
      <mmi:MD_MaintenanceFrequencyCode codeList="http://...." codeListValue="asNeeded">
          empty, or any text in any single language
      </mmi:MD_MaintenanceFrequencyCode>
    </mmi:maintenanceAndUpdateFrequency>
    <mmi:maintenanceDate>
      <cit:CI_Date>
        <cit:date>
          <gco:DateTime>2022-01-01T05:00:00-05:00</gco:DateTime>
        </cit:date>
        <cit:dateType>
          <cit:CI_DateTypeCode codeList="http://..." codeListValue="nextUpdate">
            empty, or any text in any single language
          </cit:CI_DateTypeCode>
        </cit:dateType>
      </cit:CI_Date>
    </mmi:maintenanceDate>
  </mmi:MD_MaintenanceInformation>
</mri:resourceMaintenance>
```

EXAMPLE 3: Dataset has no consistent update schedule. The next update will be on 1 January 2022 at an unspecified time.

```
<mri:resourceMaintenance>
  <mmi:MD_MaintenanceInformation>
    <mmi:maintenanceAndUpdateFrequency>
      <mmi:MD_MaintenanceFrequencyCode codeList="http://...." codeListValue="irregular"/>
```

```
        </mmi:maintenanceAndUpdateFrequency>
        <mmi:maintenanceDate>
            <cit:CI_Date>
                <cit:date>
                    <gco:Date>2022-01-01</gco:Date>
                </cit:date>
                <cit:dateType>
                    <cit:CI_DateTypeCode codeList="http://...." codeListValue="nextUpdate"/>
                </cit:dateType>
            </cit:CI_Date>
        </mmi:maintenanceDate>
    </mmi:MD_MaintenanceInformation>
</mri:resourceMaintenance>
```

S-100 – Part 18

Language Packs

Page intentionally left blank

Contents

Page intentionally left blank

18-1 Scope

This Part of S-100 details how multi-lingual support for XML elements of the framework may be implemented. A generic mechanism and structures are described for production of individual language packs which implement translations of any XML content.

This is designed to provide multi-lingual instances of XML resources which support Product Specifications for provision to end users. Implementing systems are then able to construct translated instances of those supporting resources. This Part is not specific to any one individual class of XML resource. It does not detail how multi-lingual support may be added to S-100 Product Specifications, datasets or any external resources they may reference. It provides a generic mechanism which can be applied to any XML based elements of the S-100 framework to adapt them for multi-lingual implementations.

This Part of S-100 provides the generic methodology for implementing such support and informative examples for a primary use case, the creation of multi-lingual support for S-100 Feature Catalogues.

18-2 Normative References

The following referenced documents are required for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including amendments) applies:

XML Schema : *W3C XML Schema*

XPath Specification: *W3C specification. 2017-3-21: https://www.w3.org/TR/xpath/*

ISO 639-2/T for specification of languages: *ISO 639-2/T 1998 3 letter codes*

18-3 General Description

Under S-100 a number of XML Schemas are defined. These are used in variety of contexts within the framework and by data producers and implementers to define XML content. This part introduces the concept of “language packs” to enable multi-lingual support specific to XML content.

For specific XML “source” content, a language pack is a published XML dataset which provides translations of selected elements of the source content. A language pack is, itself, XML content with a schema defined under S-100, described in section 18-6.

This mechanism enables transformation of selected elements of the source content from one language (the “source language”) to another (the “target language”).

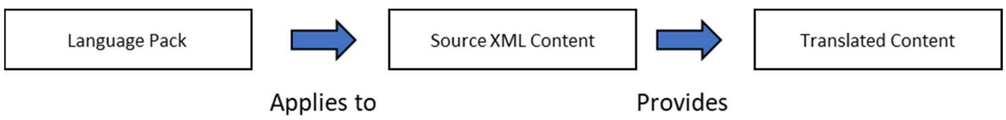


Figure 18-1 – Language pack support process

Multiple language packs may also be constructed to define multi-lingual translations of (potentially different) content from the Source XML. For example:

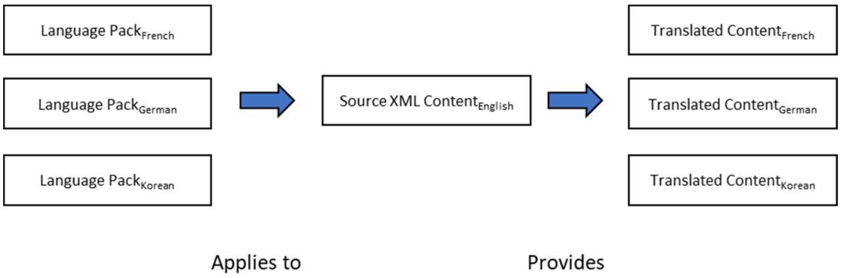


Figure 18-2 – Example multi-lingual support

Note that the language pack only provides a list of translations for specific identified elements within the source content. A version of the source content with all the relevant elements translated is not necessarily produced. The language pack and the source content are distributed and used together by implementing systems to provide multi-lingual support to an end user

For example, in Figure 18-3 below an XML Feature Catalogue and three language packs, together with a dataset are delivered in an Exchange Set to an implementing system. The dataset provides the data content and the Feature Catalogue and its language packs provides multi-lingual support for the user interface elements (for example, legends and textual descriptions of features).

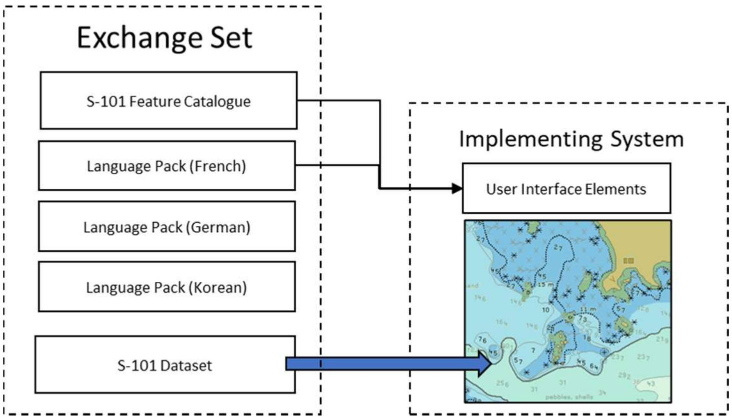


Figure 18-3 – Multi-lingual support – Exchange Set to implementing system

18-4 Language Pack Model

The model of a language pack is described in Figure 18-4 below.

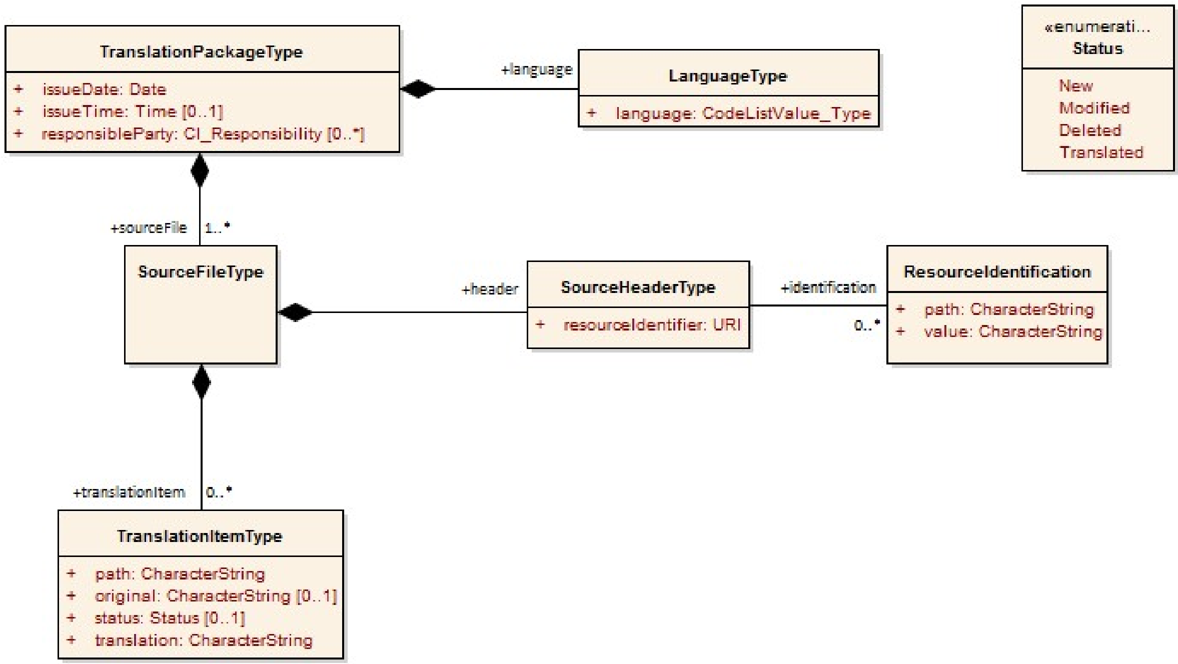


Figure 18-4 – Language pack model

The ***TranslationPackageType*** consists of a minimal set of header information and a sequence of source files elements. Each of the ***SourceFileType*** elements has a header to identify the resource and a set of items that hold the information for one text element in the resource to be translated.

Model details are in the following Tables. Note that most classes exist in a namespace **S100_LA**. This prefix is not part of the class names but logically belongs to them. Exceptions are marked in the Tables.

Page intentionally left blank

18-4.1 TranslationPackageType

Role Name	Name	Description	Mult	Type	Remarks
Class	TranslationPackageType	The main element of a language package.	-	-	-
Role	language	The language in which the translations are made		LanguageType	The type contains an attribute ***language*** that holds a language code according to ISO639-2/T (for example deu, fra)
Attribute	issueDate	The date when the language pack is issued	1	Date	
Attribute	issueTime	The time when the language pack is issued	0..1	Time	
Attribute	responsibleParty	Meta information about the responsible organisation or individual	0..*	CI_ResponsibleParty	This type conforms to ISO 19115 and is not defined in the namespace ***S100_LA.*** There may be more than one parties responsible which then have different roles. (for example custodian, translator, or publisher)
Role	sourceFile	All translation items for one source file	1..*	SourceFileType	

18-4.2 SourceFileType

Role Name	Name	Description	Mult	Type	Remarks
Class	SourceFileType	An element holding all information on elements to be translated for a single source file	-	-	The source file must be in XML format
Role	header	Information to identify the source file	1	SourceHeaderType	
Role	translationItem	A list of items each describes an element of the source file which is subject to translation	0..*	TranslationItemType	

18-4.3 SourceHeaderType

Role Name	Name	Description	Mult	Type	Remarks
Class	SourceHeaderType	Information used to identify the source file for which this translation package contains the translations for	-	-	

Attribute	resourceIdentification	The identifier of the resource	1	URI	This can be either a file URI or a hash URN Example: file:///somefile.xml
Role	identification	Information on version information located within the source file	0..*	ResourceIdentification	

18-4.4 ResourceIdentification

Role Name	Name	Description	Mult	Type	Remarks
Class	ResourceIdentification	Information where version identifiers can be found in the source (XML) file and what value is stored there	-	-	
Attribute	path	An XPath expression to the element or attribute that contains the identification information	1	CharacterString	Example: /S100FC:S100_FC_FeatureCatalogue/S100FC:versionNumber
Attribute	value	The value that must be in the specified element to match the source file	1	CharacterString	Example: 1.0.0

18-4.5 TranslationItemType

Role Name	Name	Description	Mult	Type	Remarks
Class	TranslationItemType	Information on a single translatable item in the source file	-	-	
Attribute	path	An XPath expression to the element or attribute that contains the text to be translated	1	CharacterString	
Attribute	original	The text as it appears in the source file	0..1	CharacterString	
Attribute	status	The status of the item	0..1	Status	
Attribute	translation	The translated text.	1	CharacterString	

18-4.6 Status

Item	Name	Description	Remarks
Enumeration	Status	The status of the translation item	The purpose is to support the life cycle of each translation item
Literal	New	The item is not yet translated	

Literal	Modified	The content of the item has been changed in the source file	The translation must be revisited because it may not be valid anymore
Literal	Deleted	The item does not exist in the source file anymore	
Literal	Translated	The item is translated and ready to use	

Page intentionally left blank

18-5 Language Pack Creation

18-5.1 Authoring

Language Pack authors are responsible for the definition of translated content for a particular language for selected elements of content. While some elements may always require translations (such as definitions of features or information types), others (such as language nonspecific enumeration names) may not. It is the responsibility of the language pack author to determine which elements require translation.

Translations may be revised over time and so the schema makes provision for revisions to, optionally, be contained within a language pack. These are not expected to be implemented by end user systems, however, and each language pack delivered to end user systems defines a complete translation of the required elements in the source XML into the target language.

18-5.2 Distribution

S-100 Part 17 Exchange Catalogue enables the inclusion of zero or more language packs within S-100 Exchange Sets for distribution to end users. Such language packs can be included alongside the content to which they refer or independently to supplement pre-installed content on the end user system.

Where they are delivered alongside the content to which they refer, then they shall be referenced to the content by association in the Exchange Catalogue using the mechanisms defined for supporting resources within that Part. As Exchange Set content they will be digitally signed.

There is no specific filename convention for language packs but as supporting resources they shall use the filename convention defined in Part 17.

To avoid ambiguity Exchange Sets shall only contain a single language pack for any given source resource in a given target language.

18-5.3 Implementation.

The implementer of the S-100 system is responsible for correctly interpreting delivered language packs and performing the content value substitution correctly. The implementer is also responsible for providing support for multiple language packs and any harmonization with multi-lingual support within S-100 product datasets. This can provide the end user with a harmonized experience where data content and associated user interface elements are configured using the same language specifiers.

18-6 Language Pack Schema.

18-6.1 Schema Overview.

Under this Part of S-100 a language pack is created as XML content conforming to the language pack Schema. This Schema is comprised of two components:

1. A header defining:
    a. The language implemented in the language pack.
    b. The language pack producer and time/date of issue.
2. Translation entries for one (or more) source XML resources. Each source XML resource is identified by a URI and a single element which identifies its revision. A sequence of translation entries for each source is given, each containing (as a minimum):
    a. A location specifier of an element in the source XML. This is specified using an XPath specification.
    b. The translated text of the element.

For the purposes of assisting the process of language pack creation, and maintenance the language pack Schema may also optionally contain, for each translation entry:

1. The source content, from which the translation is made.
2. A state value for the translation.

The root *translationPackage* element defines the language supported by the language pack (the destination language), the issue date and time and language pack producer. The *translationPackage* then contains any number of *sourceFile* translations providing translations into the destination language. All languages are specified using ISO639-2/T with descriptors contained in the S-100 Schema codelists. This ensures languages specified in language packs use the same descriptors as languages contained in dataset encodings.

The *sourceFile* element contains a header identifying the source xml file for which the translation is provided. Unique identification of the source resource is accomplished by matching defined XPath resources in the source with defined values in the language pack *translationPackage* identification element. This is because such unique identifiers may differ between different S-100 XML schemas and so a general path specification is used to enable unique identification of any XML content within S-100.

TranslationItems define the translations themselves. The *path* attribute in an instance of a *TranslationItemType* will point to either an element or an attribute in the source file using an XPath expression. The expression must uniquely define one element, hence a query on the XML DOM tree must return exactly one node.

For example, given the following XML source content:

```
<S100FC:S100_FC_FeatureCatalogue xmlns:S100FC=http://www.iho.int/S100FC …
…
  <S100FC:S100_FC_SimpleAttributes>
    <S100FC:S100_FC_SimpleAttribute>
      <S100FC:name>Category of Topping</S100FC:name>
      <S100FC:definition>Topping on a pizza (a maximum of four)</S100FC:definition>
      <S100FC:code>categoryOfTopping</S100FC:code>
```

A simplified tree structure of the above XML file is shown in Figure 18-5 below:

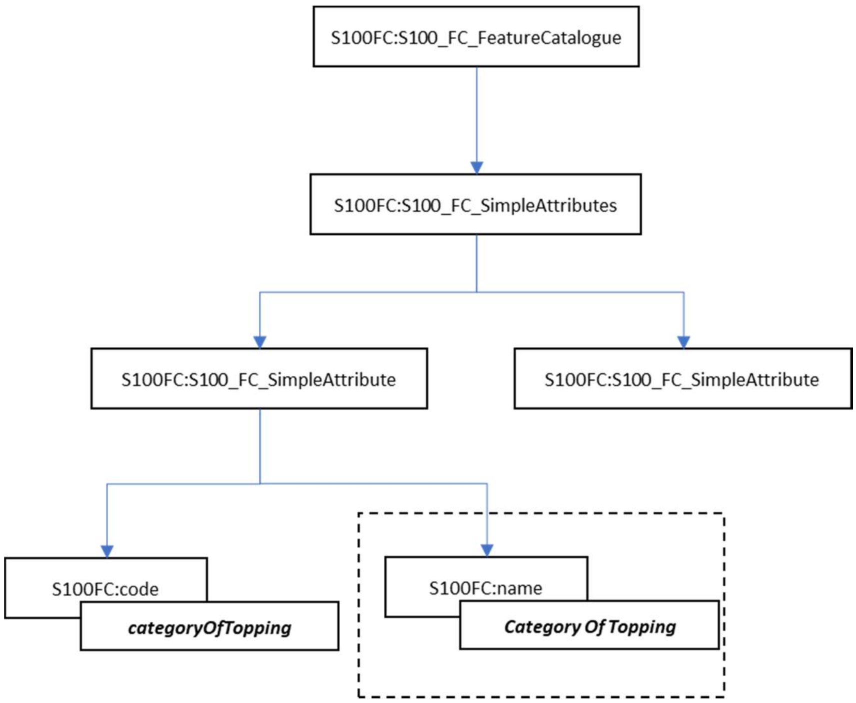


Figure 18-5 – Feature Catalogue tree structure (example)

The XPath expression:

```
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='categoryOfTopping']/S100FC:name
```

would locate the marked element by first locating elements with the name ***S100FC:S100_FC_Simple_Attribute*** with parent **S100FC:S100_FCSimpleAttributes** (and **S100FC:S100_FC_FeatureCatalog)** that has a child ***S100FC:code*** with a text of "*categoryOfTopping*". The child **S100FC:name** is then selected and then used as the reference for translation. In the general case where elements have the same code the XPath expression could be ambiguous. This problem does not arise In Feature Catalogues as the **S100FC:code** is always unique so expressions referring to codes guarantees an unambiguous path specification in the *TranslationItem*.

The TranslationItem is then encoded with a translation of the text of the element (the name) in the destination language and any optional state or revision information.

18-6.2 Schema implementation.

The Schema is defined in the namespace:

```
xmlns="http://www.iho.int/s100/la/5.0"
```

It uses types from two ISO 19115 Schemas:

```
xmlns:gco="http://standards.iso.org/iso/19115/-3/gco/1.0"
xmlns:cit="http://standards.iso.org/iso/19115/-3/cit/2.0"
```

The following settings are made for this Schema:

```
targetNamespace="http://www.iho.int/S100/la/5.0"
elementFormDefault="qualified"
attributeFormDefault="unqualified"
```

In the Schema the type **Status** is implemented as a simple type as follows:

```
<xs:simpleType name="Status">
  <xs:annotation>
    <xs:documentation>
    The status of the translation item. The purpose is manly to support the
    functionality of an translation tool.
    </xs:documentation>
  </xs:annotation>
 <xs:restriction base="xs:string">
   <xs:enumeration value="New">
     <xs:annotation>
       <xs:documentation>
       The item is new, there is no translation available yet.
       </xs:documentation>
     </xs:annotation>
   </xs:enumeration>
   <xs:enumeration value="Modified">
     <xs:annotation>
       <xs:documentation>
       The original text has been changed in the source
       </xs:documentation>
     </xs:annotation>
   </xs:enumeration>
   <xs:enumeration value="Deleted">
     <xs:annotation>
       <xs:documentation>
       The text defined by path is not longer available in the source
       document.
       </xs:documentation>
     </xs:annotation>
   </xs:enumeration>
  <xs:enumeration value="Translated">
    <xs:annotation>
      <xs:documentation>
```

```
        The item has a valid translation.
        </xs:documentation>
      </xs:annotation>
    </xs:enumeration>
  </xs:restriction>
</xs:simpleType>
```

For the definition of the language a complex type **LanguageType** is defined that is using an element of the type **gco:CodeListValue_Type** (from ISO 19115):

```
<xs:complexType name="LanguageType">
  <xs:sequence>
    <xs:element name="languageCode" type="gco:CodeListValue_Type"/>
  </xs:sequence>
</xs:complexType>
```

Each translation item is defined by the type **TranslationItemType**:

```
<xs:complexType name="TranslationItemType">
  <xs:annotation>
    <xs:documentation>
    One item to be translated. This will be a uniquely identifiable element or
    attribute in a source (XML) file
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="path" type="xs:string">
      <xs:annotation>
        <xs:documentation>
        The XPath that defines the 'source' text.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="original" type="xs:string" minOccurs="0">
      <xs:annotation>
        <xs:documentation>
        The original text as exists in the source document.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="status" type="Status" minOccurs="0">
      <xs:annotation>
        <xs:documentation>
        The status of the translation item
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="translation" type="xs:string">
      <xs:annotation>
        <xs:documentation>
        The translated text.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The type **ResourceIdentification** contains the information to identify a source file. This defines an XPath to the version element in the source and the value that this element in the source file must have:

```xml
<xs:complexType name="ResourceIdentification">
  <xs:annotation>
    <xs:documentation>
    Information to identify a specific version of an (XML) file.
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="path" type="xs:string">
      <xs:annotation>
        <xs:documentation>
        The XPath to the element or attribute that allows the unique identification
        of the source file.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="value" type="xs:string">
      <xs:annotation>
        <xs:documentation>
        The value of the element or attribute that describes the identification of
        the source file. e.g. the version or issue date
        </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The type **SourceHeaderType** contains the information on the source file as the filename and an identification mechanism:

```xml
<xs:complexType name="SourceHeaderType">
  <xs:annotation>
    <xs:documentation>
    Information to identyfing the source file. It supports mechanism to distinguish
    different versions of a source file.
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="resourceIdentifier" type="xs:anyURI"/>
    <xs:element name="identification" type="ResourceIdentification" minOccurs="0"
          maxOccurs="unbounded">
      <xs:annotation>
        <xs:documentation>
        Identification by one or more elements or attributes in the source
        file.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The type **SourceFileType** contains the header information of the file and a list of translation items for that file:

```xml
<xs:complexType name="SourceFileType">
  <xs:sequence>
    <xs:element name="header" type="SourceHeaderType"/>
    <xs:element name="translationItem" type="TranslationItemType" minOccurs="0"
    maxOccurs="unbounded">
      <xs:annotation>
        <xs:documentation>
        The list of translation items.
```

```
            </xs:documentation>
        </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The last type defined by the Schema is the type **TranslationPackageFile**:

```
<xs:complexType name="TranslationPackageType">
  <xs:sequence>
    <xs:element name="language" type="LanguageType">
        <xs:annotation>
            <xs:documentation>
            The language that will be supported by this translation
            file.
            </xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="issueDate" type="xs:date"/>
    <xs:element name="issueTime" type="xs:time" minOccurs="0"/>
    <xs:element name="responsibleParty" type="cit:CI_Responsibility_PropertyType"
     minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="sourceFile" type="SourceFileType" maxOccurs="unbounded">
        <xs:annotation>
            <xs:documentation>
            The list of source files for that this translation file provides
            translations.
            </xs:documentation>
        </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

This defines the metadata of the translation file as:

- Language;
- Issue date;
- Issue time;
- The responsible party;

and a list of source files with their translations.

Note that for the responsible party the type **cit:CI_Responsibility_PropertyType** is used.

A possible encoding would look like:

```
<S100LA:responsibleParty>
  <cit:CI_Responsibility>
    <cit:role>
      <cit:CI_RoleCode codeList="codeListLocation#CI_RoleCode"
       codeListValue="custodian">custodian</cit:CI_RoleCode>
    </cit:role>
    <cit:party>
      <cit:CI_Individual>
        <cit:name>
          <gco:CharacterString>Max Mustermann</gco:CharacterString>
        </cit:name>
      </cit:CI_Individual>
    </cit:party>
  </cit:CI_Responsibility>
</S100LA:responsibleParty>
```

As an alternative to the element <cit:CI_Individual> the element <cit:CI_Organisation> can be used if the producer is an organization rather than an individual. The predefined roles for the responsible party are defined in the appropriate codelist by the S-100 Schemas. Note that the predefined list of roles for a responsible party are. Although there is no value for '*translator*', '*contributor*' could be used for this role if required.

Finally, the Schema defines the root element of a translation file:

```
<xs:element name="translationPackage" type="TranslationPackageType">
  <xs:annotation>
    <xs:documentation>
    The root element of a translation file.
    </xs:documentation>
  </xs:annotation>
</xs:element>
```

Page intentionally left blank

Appendix 18-A
Language Packs for Feature Catalogues
(informative)

Although, as stated in the introduction to this Part, language pack production can be applied to any S-100 XML content, the primary use case is the production of language packs to support Feature Catalogues for implementing systems. This allows systems (either for end users or data producers) to use translated versions of Feature Catalogue entries such as features, information types, attributes and relationships.

In this example implementation an individual Feature Catalogue, defined by an S-100 Product Specification process contains a number of elements which could, potentially, be translated.

A set of translations is prepared of those fields required and a language pack is constructed, conformant with the language pack schema, which implements a particular language for the given Feature Catalogue.

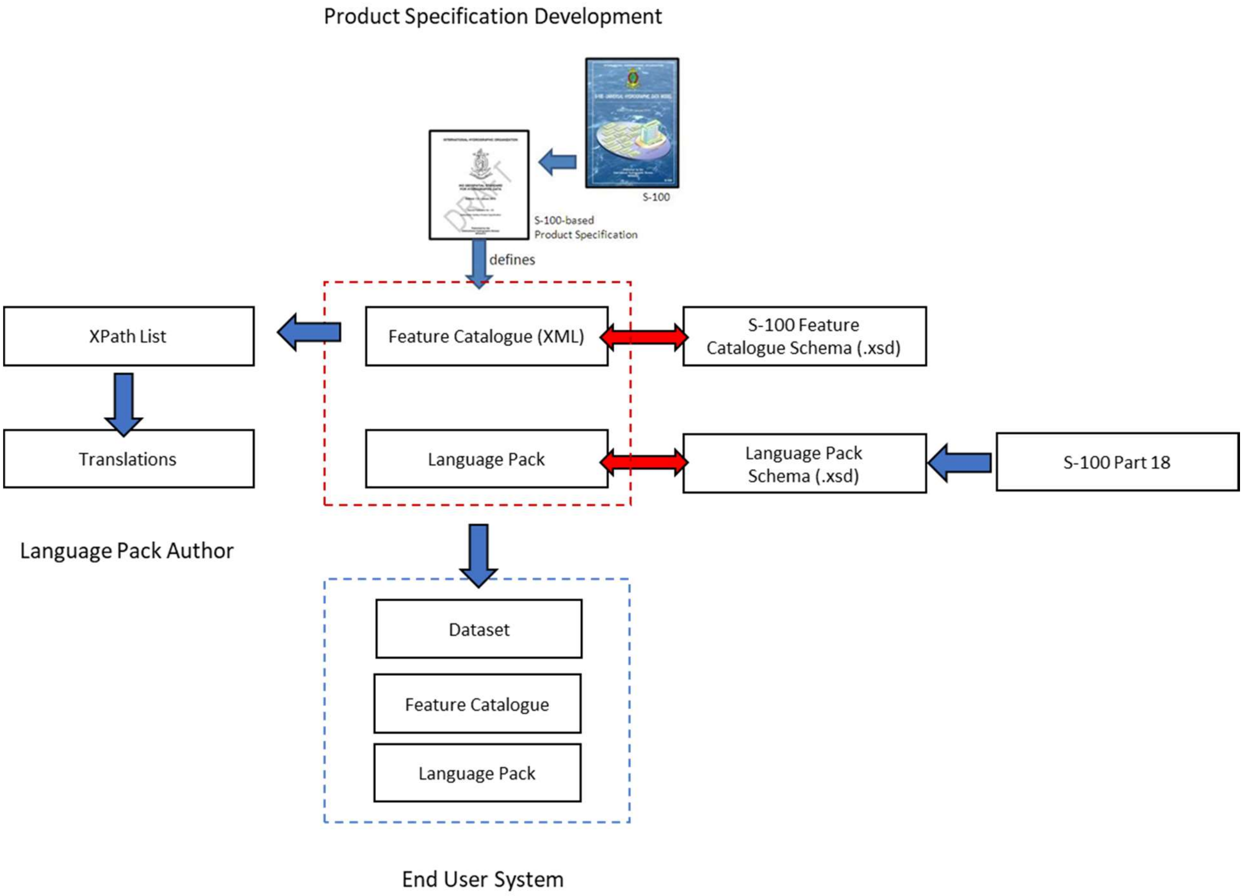


Figure 18-A-1 – Language packs in S-100

This language pack, together with the Feature Catalogue can then be distributed to end users for implementation by the end user system. The end user system is able to either construct a version of the Feature Catalogue in the language given, or interleave the translated items as required, at runtime. This is implementation-specific.

As certain elements are used as references in feature catalogues to provide bindings between entries (for example, feature attribute binding references) care should be taken to ensure such mappings remain intact, should end user systems choose to create complete translations of the Feature Catalogue at runtime. This would preclude the S100_FC_Item code field from translation as it provides the reference in attribute and sub-attribute bindings.

Language packs for feature catalogues should use the following fields for product and version identification of the source Feature Catalogue:

- `/S100FC:S100_FC_FeatureCatalogue/S100FC:name`
- `/S100FC:S100_FC_FeatureCatalogue/S100FC:versionNumber`
- `/S100FC:S100_FC_FeatureCatalogue/S100FC:versionDate`

The following fields are candidates for translation in a feature catalogue language pack (Feature Catalogue XPath locations).

General elements:

- `/S100FC:S100_FC_FeatureCatalogue/S100FC:scope`
- `/S100FC:S100_FC_FeatureCatalogue/S100FC:fieldOfApplication`

Simple Attributes:

Name, definition, and remarks are subject of translation. Note that a predicate must be used to uniquely identify the element in the source file.

In the predicate `[./S100FC:code/text()='`***code***`']`, `'`***code***`'` will be the code of the attribute; for example, `'categoryOfLight'`.

/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='***attributeCode***']/S100FC:name
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='***attributeCode***']/S100FC:definition
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='***attributeCode***']/S100FC:remarks

Listed values of enumeration attributes:

Label, definition, and remarks are subject of translation.

Note that that here two predicates are required one for the attribute code and one for the enumeration code of the listed value.

/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='***attributeCode***']/S100FC:listedValues/S100FC:listedValue[./S100FC:code/text()='***enumerationValue***']/S100FC:label
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='***attributeCode***']/S100FC:listedValues/S100FC:listedValue[./S100FC:code/text()='***enumerationValue***']/S100FC:definition
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='***attributeCode***']/S100FC:listedValues/S100FC:listedValue[./S100FC:code/text()='***enumerationValue***']/S100FC:remarks

Complex Attributes:

Similar to simple attributes.

/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_ComplexAttributes/S100FC:S100_FC_ComplexAttribute[./S100FC:code/text()='***attributeCode***']/S100FC:name
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_ ComplexAttributes /S100FC:S100_FC_ComplexAttribute[./S100FC:code/text()='***attributeCode***']/S100FC:definition
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_ ComplexAttributes /S100FC:S100_FC_ComplexAttribute[./S100FC:code/text()='***attributeCode***']/S100FC:remarks

Roles:

/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_Roles/S100FC:S100_FC_Role[./S100FC:code/text()=***'roleCode'***]/S100FC:name
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_Roles/S100FC:S100_FC_Role[./S100FC:code/text()=***'roleCode'***]/S100FC:definition
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_Roles/S100FC:S100_FC_Role[./S100FC:code/text()=***'roleCode'***]/S100FC:remarks

InformationAssociations:

/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_InformationAssociations/S100FC:S100_FC_InformationAssociation[./S100FC:code/text()=***'associationCode'***]/S100FC:name
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_InformationAssociations/S100FC:S100_FC_InformationAssociation[./S100FC:code/text()=***'associationCode'***]/S100FC:definition
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_InformationAssociations/S100FC:S100_FC_InformationAssociation[./S100FC:code/text()=***'associationCode'***]/S100FC:remarks

FeatureAssociations:

/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_FeatureAssociations/S100FC:S100_FC_FeatureAssociation[./S100FC:code/text()=***'associationCode'***]/S100FC:name
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_FeatureAssociations/S100FC:S100_FC_FeatureAssociation[./S100FC:code/text()=***'associationCode'***]/S100FC:definition
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_FeatureAssociations/S100FC:S100_FC_FeatureAssociation[./S100FC:code/text()=***'associationCode'***]/S100FC:remarks

InformationTypes:

/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_InformationTypes/S100FC:S100_FC_InformationType[./S100FC:code/text()=***'typeCode'***]/S100FC:name
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_InformationTypes/S100FC:S100_FC_InformationType[./S100FC:code/text()=***'typeCode'***]/S100FC:definition
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_InformationTypes/S100FC:S100_FC_InformationType[./S100FC:code/text()=***'typeCode'***]/S100FC:remarks

FeatureTypes:

/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_FeatureTypes/S100FC:S100_FC_FeatureType[./S100FC:code/text()=***'typeCode'***]/S100FC:name
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_FeatureTypes/S100FC:S100_FC_FeatureType[./S100FC:code/text()=***'typeCode'***]/S100FC:definition
/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_FeatureTypes/S100FC:S100_FC_FeatureType[./S100FC:code/text()=***'typeCode'***]/S100FC:remarks

As stated in Section18-5, Feature Catalogues may produce translations of some or all of these fields, depending on the preference of the language pack author.

An example of some of these fields are shown in the extract from an S-101 feature catalogue below:

```
    <S100FC:S100_FC_SimpleAttribute>
      <S100FC:name>Category of Light</S100FC:name>
      <S100FC:definition>Classification of different light types.</S100FC:definition>
      <S100FC:code>categoryOfLight</S100FC:code>
      <S100FC:remarks>All lights are considered to be marine lights unless the category of light indicates otherwise.</S100FC:remarks>
      <S100FC:alias>CATLIT</S100FC:alias>
      <S100FC:definitionReference>
        <S100FC:sourceIdentifier>88</S100FC:sourceIdentifier>
        <S100FC:definitionSource ref="IHOREG"/>
      </S100FC:definitionReference>
      <S100FC:valueType>enumeration</S100FC:valueType>
      <S100FC:listedValues>
        <S100FC:listedValue>
          <S100FC:label>Leading Light</S100FC:label>
          <S100FC:definition>A light associated with other lights so as to form a leading line to be followed.</S100FC:definition>
          <S100FC:code>4</S100FC:code>
          <S100FC:definitionReference>
            <S100FC:sourceIdentifier>863</S100FC:sourceIdentifier>
            <S100FC:definitionSource ref="IHOREG"/>
          </S100FC:definitionReference>
        </S100FC:listedValue>
        <S100FC:listedValue>
          <S100FC:label>Aero Light</S100FC:label>
          <S100FC:definition>An aero light is established for aeronautical navigation and may be of higher power than marine lights and visible from well offshore.</S100FC:definition>
          <S100FC:code>5</S100FC:code>
          <S100FC:definitionReference>
            <S100FC:sourceIdentifier>864</S100FC:sourceIdentifier>
            <S100FC:definitionSource ref="IHOREG"/>
          </S100FC:definitionReference>
        </S100FC:listedValue>
```

This section of the S-101 Feature Catalogue would result in the following translations included in a German S-101 feature catalogue language pack:

Table 18-A-1 – Language pack translation (examples)

Element	English Text	German Text
Name[code= categoryOfLight]	Category of Light	Kategorie des Leuchtfeuers
Definition[code= categoryOfLight]	Classification of different light types	Die Systematik der unterschiedlichen Leuchtfeuerarten
Remarks[code= categoryOfLight]	All lights are considered to be marine lights unless the category of light indicates otherwise	Alle Leuchtfeuer werden als maritime Leuchtfeuer betrachtet, solange die Kategorie nichts anderes angibt.
listedValue/label[code=4]	Leading Light	Richtfeuer
listedValue/definition[code=4]	A light associated with other lights so as to form a leading line to be followed.	Ein Feuer, dass zusammen mit anderen Feuern eine Linie bestimmt, der gefolgt werden muss.
listedValue/label[code=5]	Aero Light	Luftfahrtfeuer
listedValue/definition[code=5]	An aero light is established for aeronautical navigation and may be of higher power than marine lights and visible from well offshore	Feuer für Luftfahrtsnavigation. Hat eventuell eine höhere Leuchtstärke als maritime

		Leuchtfeuer und kann weit vor der Küste sichtbar sein.

The values in the table would be encoded in the German S-101 language pack, as demonstrated in the following XML fragment.

Note that not all items are translated in this example. For those that are translated the status is set to 'Translated'. All other items have still the status 'New' meaning not yet translated.

```xml
<S100LA:translationItem>

<S100LA:path>/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='categoryOfLight']/S100FC:name</S100LA:path>
    <S100LA:original>Category of Light</S100LA:original>
    <S100LA:status>Translated</S100LA:status>
    <S100LA:translation>Kategorie des Leuchtfeuers</S100LA:translation>
  </S100LA:translationItem>
  <S100LA:translationItem>

<S100LA:path>/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='categoryOfLight']/S100FC:definition</S100LA:path>
    <S100LA:original>Classification of different light types.</S100LA:original>
    <S100LA:status>Translated</S100LA:status>
    <S100LA:translation>Die Systematik der unterschiedlichen Leuchtfeuerarten </S100LA:translation>
  </S100LA:translationItem>
  <S100LA:translationItem>

<S100LA:path>/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='categoryOfLight']/S100FC:remarks</S100LA:path>
    <S100LA:original>All lights are considered to be marine lights unless the category of light indicates
otherwise.</S100LA:original>
    <S100LA:status>New</S100LA:status>
    <S100LA:translation />
  </S100LA:translationItem>
  <S100LA:translationItem>

<S100LA:path>/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='categoryOfLight']/S100FC:listedValues/S100FC:listedValue[./S100FC:code/text()='4']/S100FC:label</S100LA:path>
    <S100LA:original>Leading Light</S100LA:original>
    <S100LA:status>New</S100LA:status>
    <S100LA:translation />
  </S100LA:translationItem>
  <S100LA:translationItem>

<S100LA:path>/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='categoryOfLight']/S100FC:listedValues/S100FC:listedValue[./S100FC:code/text()='4']/S100FC:definition</S100LA:path>
    <S100LA:original>A light associated with other lights so as to form a leading line to be
followed.</S100LA:original>
    <S100LA:status>New</S100LA:status>
    <S100LA:translation />
  </S100LA:translationItem>
  <S100LA:translationItem>

<S100LA:path>/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='categoryOfLight']/S100FC:listedValues/S100FC:listedValue[./S100FC:code/text()='5']/S100FC:label</S100LA:path>
    <S100LA:original>Aero Light</S100LA:original>
    <S100LA:status>New</S100LA:status>
    <S100LA:translation />
  </S100LA:translationItem>
```

```
  <S100LA:translationItem>
<S100LA:path>/S100FC:S100_FC_FeatureCatalogue/S100FC:S100_FC_SimpleAttributes/S100FC:S100_FC_SimpleAttribute[./S100FC:code/text()='categoryOfLight']/S100FC:listedValues/S100FC:listedValue[./S100FC:code/text()='5']/S100FC:definition</S100LA:path>
    <S100LA:original>An aero light is established for aeronautical navigation and may be of higher power than marine lights and visible from well offshore.</S100LA:original>
    <S100LA:status>New</S100LA:status>
    <S100LA:translation />
  </S100LA:translationItem>
```

This language pack would be named according to the naming convention in S-100 Part 17 and would support a Feature Catalogue using the reference structure defined.

S-100 – Annex A

Terms and Definitions

Page intentionally left blank

ANNEX A

Terms and definitions

For the purposes of this document, the following terms and definitions apply:

2.5 dimension
two-dimensional topology used with a three-dimensional coordinate system constrained to a two-dimensional manifold
[ISO 19107]

abstract class
an abstract class defines a polymorphic object class which cannot be instantiated
[ISO 19103]

accuracy
closeness of agreement between a test result and the accepted reference values [ISO 3534-1]
NOTE A test result can be from an observation or measurement.

addition
insertion of an item into the register
[ISO 19135]

affine coordinate system
coordinate system in Euclidean space with straight axes that are not necessarily mutually perpendicular
[ISO 19111]

aggregation
special form of association that specifies a whole-part relationship between the aggregate (whole) and a component part (see composition)
[ISO 19103]

annotation
any marking on illustrative material for the purpose of clarification
NOTE Numbers, letters, **symbols**, and signs are examples of annotation.
[ISO 19117:2012 (E), 4.1]

application programming interface
one implementation of the required application services as defined in IEC 61162-401
NOTE One API from one manufacturer may be different from another API, although the basic functionality is the same.

application schema
conceptual schema for data required by one or more applications
[ISO 19101]

association
semantic relationship between two or more classifiers that specifies connections among their instances [ISO 19103]
NOTE A binary association is an association among exactly two classifiers (including the possibility of an association from a classifier to itself).

attribute
(1) named property of an entity [ISO/IEC 2382-17:1999]
NOTE Describes a geometrical, topological, thematic, or other characteristic of an entity.
(2) feature within a classifier that describes a range of values that instances of the classifier may hold
NOTE 1 An attribute is semantically equivalent to a composition association; however, the intent and usage is normally different.

NOTE 2 "Feature" used in this definition is the UML meaning of the term and is not meant as defined in clause 4 of this part.

A-profile
communication protocol supplying application services (see OSI 5 to 7)

band
range of wavelengths of electromagnetic radiation that produce a single response by a sensing device [ISO/TS 19101-2:2008]

base standard
ISO geographic information standard or other information technology standard that is used as a source from which a profile may be constructed [ISO 19106:2005]

BLOB
(Binary Large Object) a collection of binary data stored as a single entity

boundary
set that represents the limit of an entity [ISO 19107]
NOTE Boundary is most commonly used in the context of geometry, where the set is a collection of points or a collection of objects that represent those points.

cartesian coordinate system
coordinate system which gives the position of points relative to *n* mutually perpendicular axes [ISO 19111]

character
in online data exchange: an octet containing a code from the set defined in ISO/IEC 8859-1. The null character (octet containing all zero bits) may have special meaning

chart window
the portion of the screen that is displaying chart feature information in a graphical representation similar to that of a nautical chart

clarification
non-substantive change to a register item [ISO 19135]
NOTE A non-substantive change does not change the semantics or technical meaning of the item. A clarification does not result in a change to the registration status of the register item.

class
description of a set of objects that share the same attributes, operations, methods, relationships, and semantics [ISO/TS 19103:2005]
NOTE 1 A class represents a concept within the system being modelled. Depending on the kind of model, the concept may be real-world (for an analysis model), or it may also contain algorithmic and computer implementation concepts (for a design model). A classifier is a generalization of class that includes other class-like elements, such as data type, actor and component.
NOTE 2 A class may use a set of interfaces to specify collections of operations it provides to its environment. See: interface.

classification
the process of determining the appropriate type within a feature catalogue for a particular real world feature, including consideration of data quality

classifier
mechanism that describes behavioural and structural features [ISO 19103]
NOTE Classifiers include interfaces, classes, datatypes, and components.

client
a technical entity (for example: Device, Program) which uses a service

co-attribute (of an attribute)
a sub-attribute of the same complex attribute; an attribute of the same feature or information type

code list
value domain including a code for a permissible value
[ISO 19136]

colour token
an identifier for a colour in a palette

complex attribute
an aggregation of other attributes which may be either simple or complex
[ISO 19109]

composite curve
sequence of curves such that each curve (except the first) starts at the end point of the previous curve in the sequence
[ISO 19107]

composition
form of aggregation association with strong ownership and coincident lifetime as part of the whole
[ISO 19103]
NOTE Parts with non-fixed multiplicity may be created after the composite itself, but once created they live and die with it (that is, they share lifetimes). Such parts can also be explicitly removed before the death of the composite. Composition may be recursive. Synonym: composite aggregation.

compound coordinate reference system
coordinate reference system using at least two independent coordinate reference systems
[ISO 19111]

concatenated coordinate operation
coordinate operation consisting of sequential application of multiple coordinate operations
[ISO 19111]

conceptual model
model that defines the concepts of a universe of discourse
[ISO 19101]

conceptual schema
formal description of a conceptual model
[ISO 19101]

conformance
fulfilment of specified requirements
[ISO 19105]

continuous coverage
coverage that returns different values for the same feature attribute at different direct positions within a single geometric object in its spatiotemporal domain [ISO 19123]
NOTE Although the spatiotemporal domain of a continuous coverage is ordinarily bounded in terms of its spatial extent, it can be subdivided into an infinite number of direct positions.

control body
group of technical experts that makes decisions regarding the content of a register
[ISO 19135]

coordinate
one of a sequence of *n* numbers designating the position of a point in N-dimensional space
[ISO 19111]
NOTE The numbers must be qualified by units.

coordinate conversion
coordinate operation in which both coordinate reference systems are based on the same datum
[ISO 19111]

coordinate operation
change of coordinates, based on a one-to-one relationship, from one coordinate reference system to another
[ISO 19111]

coordinate reference system
a coordinate system that is related to the real world by a datum [ISO 19111]
NOTE For geodetic and vertical datums, it will be related to the Earth.

coordinate system
set of mathematical rules for specifying how coordinates are to be assigned to points
[ISO 19111]

coordinate transformation
coordinate operation in which the two coordinate reference systems are based on different datums
[ISO 19111]

coordinate tuple
ordered list of coordinates

coverage
feature that acts as a function to return values from its range for any direct position within its spatial, temporal or spatiotemporal domain [ISO 19123:2005]
EXAMPLE Examples include a raster image, polygon overlay, or digital elevation matrix

coverage geometry
configuration of the domain of a coverage described in terms of coordinates
[ISO 19123]

curve
1-dimensional geometric primitive, representing the continuous image of a line [ISO 19107]
NOTE The boundary of a curve is the set of points at either end of the curve. If the curve is a cycle, the two ends are identical, and the curve (if topologically closed) is considered to not have a boundary. The first point is called the start point, and the last is the end point. Connectivity of the curve is guaranteed by the "continuous image of a line" clause. A topological theorem states that a continuous image of a connected set is connected.

curve segment
1-dimensional geometric object used to represent a continuous component of a curve using homogeneous interpolation and definition methods [ISO 19107]
NOTE The geometric set represented by a single curve segment is equivalent to a curve.

cycle
spatial object without a boundary [ISO 19107]
NOTE Cycles are used to describe boundary components (see ring). A cycle has no boundary because it closes on itself, but it is bounded (that is, it does not have infinite extent). A circle or a sphere, for example, has no boundary, but is bounded.

data
reinterpretable representation of information in a formalised manner suitable for communication, interpretation, or processing

[ISO/IEC 2382-1:1993]

data capture and classification guide
instructions describing the data capturing process and the process of classification

data compaction
reduction of the number of data elements, bandwidth, cost, and time for the generation, transmission, and storage of data without loss of information by eliminating unnecessary redundancy, removing irrelevancy, or using special coding [ANS T1.523-2001]
NOTE Whereas data compaction reduces the amount of data used to represent a given amount of information, data compression does not.

data compression
compression: reduction in the number of bits used to represent source image data
[ISO 10918-1 (JPEG Part 1)]
NOTE Data compression does not reduce the amount of data used to represent a given amount of information, whereas data compaction does. Both data compression and data compaction result in the use of fewer data elements for a given amount of information.

data marshalling
defines a transmission format for data records that is independent of computer architecture, network particulars, compilers and programming languages. Data marshalling routines convert between this transport format and internal data representations used in different modules

data product
a dataset or dataset series that conforms to a data product specification
[ISO 19131]

data product specification
a detailed description of a dataset or dataset series together with additional information that will enable it to be created, supplied to and used by another party [ISO 19131]
NOTE A data product specification provides a description of Hydrographic Concepts and a specification for mapping the universe of discourse to a dataset. It may be used for production, sales, end-use or other purposes.

data quality date
date or range of dates on which a data quality measure is applied
[ISO 19113]

data quality element
quantitative component documenting the quality of a dataset [ISO 19101]
NOTE The applicability of a data quality element to a dataset depends on both the dataset's content and its product specification, the result being that all data quality elements may not be applicable to all datasets

data quality evaluation procedure
operations used in applying and reporting quality evaluation methods and their results
[ISO 19113]

data quality measure
evaluation of a data quality subelement [ISO 19113]
EXAMPLE The percentage of the values of an attribute that are correct.

data quality overview element
non-quantitative component documenting the quality of a dataset [ISO 19101]
NOTE Information about the purpose, usage and lineage of a dataset is non-quantitative quality information.

data quality result
value or set of values resulting from applying a data quality measure or the outcome of evaluating the

obtained value or set of values against a specified conformance quality level [ISO 19113]
EXAMPLE A data quality result of “90” with a data quality value type of “percentage” reported for the data quality element and its data quality subelement “completeness, commission” is an example of a value resulting from applying a data quality measure to a data specified by a data quality scope. A data quality result of “true” with a data quality value type of “Boolean variable” is an example of comparing the value (90) against a specified acceptable conformance quality level (85) and reporting an evaluation of a kind, pass or fail.

data quality scope
extent or characteristic(s) of the data for which quality information is reported [ISO 19113]
NOTE A data quality scope for a dataset can comprise a dataset series to which the dataset belongs, the dataset itself, or a smaller grouping of data located physically within the dataset sharing common characteristics. Common characteristics can be an identified feature type, feature attribute, or feature relationship; data collection criteria; original source; or a specified geographic or temporal extent.

data quality subelement
component of a data quality element describing a certain aspect of that data quality element [ISO 19113]

data quality value type
value type for reporting a data quality result [ISO 19113]
EXAMPLE “boolean variable”, “percentage”, “ratio”
NOTE A data quality value type is always provided for a data quality result.

data quality value unit
value unit for reporting a data quality result [ISO 19113]
EXAMPLE “metre”
NOTE A data quality value unit is provided only when applicable for a data quality result.

data type
specification of a value domain with operations allowed on values in this domain [ISO/TS 19103:2005]
EXAMPLE Integer, Real, Boolean, String, DirectPosition and Date
NOTE Data types include primitive predefined types and user-definable types.
NOTE A data type is identified by a term, for example Integer

dataset
identifiable collection of data [ISO 19115]
NOTE A dataset may be a smaller grouping of data which, though limited by some constraint such as spatial extent or feature type, is located physically within a larger dataset. Theoretically, a dataset may be as small as a single feature or feature attribute contained within a larger dataset. A hardcopy map or chart may be considered a dataset.

dataset series
collection of datasets sharing the same product specification [ISO 19115:2003]

datum
parameter or set of parameters that define the position of the origin, the scale, and the orientation of a coordinate system [ISO 19113]

dependency
relationship between two modelling elements, in which a change to one modelling element (the independent element) will affect the other modelling element (the dependent element)

direct position
position described by a single set of coordinates within a coordinate reference system [ISO 19107]

discrete coverage
coverage that returns the same feature attribute values for every direct position within any single geometric object in its spatiotemporal domain [ISO 19123]
NOTE The spatiotemporal domain of a discrete coverage consists of a finite set of geometric objects.

domain
well-defined set [ISO/TS 19103:2005]
NOTE Domains are used to define the domain set and range set of attributes, operators and functions.

domain specific catalogue functions
scripting functions provided within a scripting catalogue which are not part of the standard catalogue functions

domain specific functions
all scripting functions which are defined outside S-100 Part 13. The union of Domain Specific Host Functions and Domain Specific Catalogue Functions

domain specific host functions
scripting functions provided by a host to support domain-specific functionalities

ellipsoid
surface formed by the rotation of an ellipse about a main axis [ISO 19111]
Mathematically it is expressed in Cartesian coordinates as: $\frac{x^2+y^2}{a^2}+\frac{z^2}{b^2}=1$
Where 'a' is the semi-major axis and 'b' is the semi-minor axis. The latter is the rotation axis, such ellipsoids are also called oblate spheroids.

ellipsoidal coordinate system
coordinate system in which position is specified by geodetic latitude, geodetic longitude and (in the three dimensional case) ellipsoidal height
[ISO 19111]

ellipsoidal height
distance of a point from the ellipsoid measured along the perpendicular from the ellipsoid to this point; positive if upwards or outside of the ellipsoid
[ISO 19111]

encoding
conversion of data into a series of codes
[ISO 19118]

end point
last point of a curve
[ISO 19107]

event
action which occurs at an instant
[ISO 19108:2002]

exterior
difference between the universe and the closure [ISO 19107]
NOTE The concept of exterior is applicable to both topological and geometric complexes.

face
2-dimensional topological primitive [ISO 19107]
NOTE The geometric realization of a face is a surface. The boundary of a face is the set of directed edges within the same topological complex that are associated to the face via the boundary relations. These can be organized as rings.

feature
abstraction of real world phenomena [ISO 19101:2003]
NOTE A feature may occur as a type or an instance. Feature type or feature instance should be used when only one is meant.
EXAMPLE The phenomenon named 'Eiffel Tower' may be classified with other phenomena into a feature type 'tower'.

feature association
relationship that links instances of one feature type with instances of the same or a different feature type
[ISO 19110]

feature attribute
characteristic of a feature [ISO 19101]
NOTE A feature attribute type has a name, a data type and a domain associated to it. A feature attribute instance has an attribute value taken from the value domain of the feature attribute type.
EXAMPLE 1 A feature attribute named ‘colour’ may have an attribute value ‘green’ which belongs to the data type ‘text’.
EXAMPLE 2 A feature attribute named ‘length’ may have an attribute value ’82,4’ which belongs to the data type ‘real’.

feature catalogue
a catalogue containing definitions and descriptions of the feature types, feature attributes, and feature associations occurring in one or more sets of geographic data
[ISO 19110]

feature portrayal function
function that maps a geographic feature to a **symbol**
[ISO 19117:2012 (E), 4.10]

field
a named collection of labeled *subfield(s)*
EXAMPLE IHO attribute label/code and IHO Attribute Value are collected into a field named Feature Record Attribute.

flattening
ratio of the difference between the semi-major (a) and semi-minor axis (b) of an ellipsoid to the semi-major axis $f = \frac{a-b}{a}$
[ISO 19111]

function
rule that associates each element from a domain (source, or domain of the function) to a unique element in another domain (target, co-domain, or range) [ISO 19107]
NOTE The range is defined by another domain.

generalization
taxonomic relationship between a more general element and a more specific element [ISO 19103]
NOTE The more specific element is fully consistent with the more general element and contains additional information. An instance of the more specific element may be used where the more general element is allowed. See: inheritance.

geodetic coordinate reference system
coordinate reference system based on a geodetic datum
[ISO 19111]

geodetic datum
datum describing the relationship of a 2- or 3-dimensional coordinate system to the Earth
[ISO 19111]

geodetic latitude
angle from the equatorial plane to the perpendicular to the ellipsoid through a given point, northwards treated as positive
[ISO 19111]

geodetic longitude
angle from the prime meridian plane to the meridian plane of a given point, eastward treated as positive
[ISO 19111]

geographic information
Information concerning phenomena implicitly or explicitly associated with a location relative to the Earth
[ISO 19101:2003]

geolocation information
information used to determine geographic location corresponding to image location

geometric aggregate
collection of geometric objects that has no internal structure
[ISO 191107]

geometric boundary
boundary represented by a set of geometric primitives of smaller geometric dimension that limits the extent of a geometric object
[ISO 19107]

geometric complex
set of disjoint geometric primitives where the boundary of each geometric primitive can be represented as the union of other geometric primitives of smaller dimension within the same set
[ISO 19107]
NOTE The geometric primitives in the set are disjoint in the sense that no direct position is interior to more than one geometric primitive. The set is closed under boundary operations, meaning that for each element in the geometric complex, there is a collection (also a geometric complex) of geometric primitives that represents the boundary of that element. Recall that the boundary of a point (the only 0D primitive object type in geometry) is empty. Thus, if the largest dimension geometric primitive is a surface (2-D), the composition of the boundary operator in this definition terminates after at most two steps. It is also the case that the boundary of any object is a cycle.

geometric dimension
largest number n such that each direct position in a geometric set can be associated with a subset that has the direct position in its interior and is similar (isomorphic) to R^n, Euclidean n-space
[ISO 19107]

geometric object
spatial object representing a set of direct positions [ISO 19107]
NOTE A geometric object consists of a geometric primitive, a collection of geometric primitives, or a geometric complex treated as a single entity. A geometric object may be the spatial characteristics of an object such as a feature or a significant part of a feature.

geometric primitive
geometric object representing a single, connected, homogeneous element of geometry [ISO 19107]
NOTE Geometric primitives are non-decomposed objects that present information about geometric configuration. They include points, curves, surfaces, and solids.

geometry value object
object composed of a set of geometry value pairs such that the geometric object elements of the geometry value pairs are elements of a larger geometric object
[ISO 19123]

georectified
corrected for positional displacement with respect to the surface of the earth

georeferencing
process of determining the relation between the position of data in the image coordinates and its geographic or map location

grid
network composed of two or more sets of curves in which the members of each set intersect the members of the other sets in an algorithmic way [ISO 19123:2005]
NOTE The curves partition a space into grid cells.

grid coordinate system
coordinate system in which position is specified relative to the intersection of curves

grid coordinates
sequence of two or more numbers specifying a position with respect to its location on a grid

grid point
point located at the intersection of two or more curves in a grid
[ISO 19123]

gridded data
data whose attribute values are associated with positions on a grid coordinate system

ground control point
point on the earth that has an accurately known geographic position

host
the environment hosting a Lua interpreter. Typically the host is an application which uses one or more S-100 based products, such as an ECDIS

host functions
the scripting functions provided by a host. The union of the Standard Host Functions and the Domain Specific Host Functions

human readable
a representation of information that can be naturally read by humans

identifier
a linguistically independent sequence of characters capable of uniquely and permanently identifying that with which it is associated
[adapted from ISO/IEC 11179-3:2003]

image
gridded coverage whose attribute values are a numerical representation of a physical parameter
NOTE The physical parameters are the result of measurement by a sensor or a prediction from a model.

image coordinate reference system
coordinate reference system based on an image datum

image datum
datum which defines the relationship of a coordinate system to an image

imagery
representation of phenomena as images produced by electronic and/or optical techniques
[ISO 19101-2:2008]
NOTE In this part of ISO 19115, it is assumed that the objects and phenomena have been sensed or

detected by camera, infrared and multispectral scanners, radar and photometers, or other remote sensing instruments and devices.

inheritance
mechanism by which more specific elements incorporate structure and behaviour of more general elements related by behaviour [ISO 19103]
NOTE See generalization.

instance
entity to which a set of operations can be applied and which has a state that stores the effects of the operations [ISO 19103]
NOTE See: object.

interior
set of all direct positions that are on a geometric object but which are not on its boundary [ISO 19107]
NOTE The interior of a topological object is the homomorphic image of the interior of any of its geometric realizations. This is not included as a definition because it follows from a theorem of topology.

ISO/IEC 8211 record
an ISO/IEC 8211 implementation of a S-57 *record* and which comprises one or more *fields*

label
an ISO/IEC 8211 implementation concept used to identify the *subfield*

LIDAR
system consisting of 1) a photon source (frequently, but not necessarily, a laser), 2) a photon detection system, 3) a timing circuit, and 4) optics for both the source and the receiver that uses emitted laser light to measure ranges to and/or properties of solid objects, gases, or particulates in the atmosphere
NOTE Time of flight (TOF) LIDARs use short laser pulses and precisely record the time each laser pulse was emitted and the time each reflected return(s) is received in order to calculate the distance(s) to the scatterer(s) encountered by the emitted pulse. For topographic LIDAR, these time-of-flight measurements are then combined with precise platform location/attitude data along with pointing data to produce a three-dimensional product of the illuminated scene of interest. [ISO/TS 19130-2:2014, [4.40], via ISO Multilingual Glossary of Terms].

machine readable
a representation of information that can be processed by computers

map projection
coordinate conversion from an ellipsoidal coordinate system to a plane
[ISO 19111]

meridian
intersection of an ellipsoid by a plane containing the shortest axis of the ellipsoid
[ISO 19111]

message
a fixed format sequence of data that are exchanged

metadata
data about data
[ISO 19115:2005]

metadata element
discrete unit of metadata
NOTE Metadata elements are unique within a metadata entity.
NOTE Equivalent to an attribute in UML terminology.
[ISO 19115:2005]

metadata entity
set of metadata elements describing the same aspect of data
NOTE May contain one or more metadata entities.
NOTE Equivalent to a class in UML terminology
[ISO 19115:2005]

metadata section
subset of metadata which consists of a collection of related metadata entities and metadata elements
NOTE Equivalent to a package in UML terminology
[ISO 19115:2005]

metamodel
model that defines the language for expressing a model

model
abstraction of some aspects of universe of discourse [ISO 19101]
NOTE A semantically complete abstraction of a system.

modification
a substantive semantic change to a register item
[ISO 19135]

multiplicity
specification of the number of possible occurences of a property, or the number of allowable elements that may participate in a given relationship [ISO 19103]
EXAMPLES 1..* (one to many) , 1 (exactly one), 0..1 (zero or one).

navigation display (or **navigation screen**)
the ECDIS or INS display(s) which is being used for route monitoring or collision avoidance
NOTE Displays showing only non-S-100 information (e.g., radar, meteorological information) are excluded from the scope of this document.

object
entity with a well-defined boundary and identity that encapsulates state and behaviour
NOTE State is represented by attributes and relationships, behaviour is represented by operations, methods, and state machines. An object is an instance of a class. See: class, instance.

operation
in online data exchange: a function(s) needed on the server and/or client in an Online Data Exchange service in order to correctly accomplish the intended service

package
general purpose mechanism for organizing elements into groups [ISO 19103]
NOTE Packages may be nested within other packages. Both model elements and diagrams may appear in a package.

palette
collection of colours defined in CIE and/or sRGB colour space and identified by a colour token

pass
single instance of a remote, mobile measuring system going by a target of interest
NOTE In this part of ISO 19115, the measuring system will usually be a remote sensing platform. In a navigation context, the measuring system might be a GPS satellite.

pixel
smallest element of a digital image to which attributes are assigned [ISO 19129]
NOTE It is the smallest unit of display for a visible image.

platform
structure which supports a sensor, or sensors

point
0-dimensional geometric primitive, representing a position [ISO 19107]
NOTE The boundary of a point is the empty set.

point coverage
coverage that has a spatial domain composed of points
[ISO 19123]

polarisation
restricting radiation, especially light, vibrations to a single plane

portrayal
presentation of information to humans [ISO 19117]
NOTE Within the scope of this International Standard portrayal is restricted to the portrayal of geographic information.

portrayal catalogue
collection of defined **portrayals** for a feature catalogue
NOTE Content of a portrayal catalogue includes **portrayal functions**, **symbols**, and **portrayal context**.
[ISO 19117:2012 (E), 4.21]

portrayal context
circumstances, imposed by factors extrinsic to a geographic dataset, that affect the **portrayal** of that dataset
NOTE Portrayal context can influence the selection of **portrayal functions** and construction of **symbols**.
EXAMPLE Factors contributing to portrayal context can include the proposed display or map scale, the viewing conditions (day/night/dusk), and the display orientation requirements (north not necessarily at the top of the screen or page) among others.
[ISO 19117:2012 (E), 4.22]

portrayal function
function that maps geographic features to **symbols**
NOTE **Portrayal** functions can also include parameters and other computations that are not dependent on geographic feature properties.
[ISO 19117:2012 (E), 4.23]

portrayal rule
specific type of **portrayal function** expressed in a declarative language
NOTE A declarative language is rule-based and includes decision and branching statements.
[ISO 19117:2012 (E), 4.25]

pre-order Traversal Sequence
representation of the order in which information, in a tree structure diagram, must be interpreted. The sequence is extremely important and inviolate as there is no other explicit method of specifying the interfield (parent/child) relationships within the ISO/IEC 8211 data records.

prime meridian
meridian from which the longitudes of other meridians are quantified
[ISO 19111]

profile
set of one or more base standards or subsets of base standards, and, where applicable, the identification of chosen clauses, classes, options and parameters of those base standards, that are necessary for accomplishing a particular function
NOTE A profile is derived from base standards so that by definition, conformance to a profile is conformance to the base standards from which it is derived.
[ISO 19106:2005]

projected coordinate reference system
coordinate reference system derived from a two-dimensional geodetic coordinate reference system by applying a map projection
[ISO 19111]

quadtree
expression of a two-dimensional object as a tree structure of quadrants, which are formed by recursively subdividing each non-homogeneous quadrant until all quadrants are homogeneous with respect to a selected characteristic, or until a predetermined cut-off depth is reached
[ISO 2382]

quality
totality of characteristics of a product that bear on its ability to satisfy stated and implied needs
[ISO 19113]

range <coverage>
set of values associated by a function with the elements of the spatiotemporal domain of a coverage
[ISO 19123]

raster
usually rectangular pattern of parallel scanning lines forming or corresponding to the display on a cathode ray tube [ISO 19123]
NOTE A raster is a type of grid.

realization
relationship between a specification and its implementation [ISO 19103]
NOTE An indication of the inheritance of behaviour without the inheritance of structure.

record
finite, named collection of related items (objects or values) [ISO 19107]
NOTE Logically, a record is a set of pairs <name, item >.

rectified grid
grid for which there is a linear relationship between the grid coordinates and the coordinates of an external coordinate reference system [ISO 19123]
NOTE If the coordinate reference system is related to the earth by a datum, the grid is a georectified grid.

referenceable grid
grid associated with a transformation that can be used to convert grid coordinate values to values of coordinates referenced to an external coordinate reference system [ISO 19123]
NOTE If the coordinate reference system is related to the earth by a datum, the grid is a georeferenceable grid.

register
set of files containing identifiers assigned to items with descriptions of the associated items
[ISO 19135]
NOTE Descriptions may consist of many types of information, including names, definitions and codes.

register manager
organization to which management of a register has been delegated by the register owner
[ISO 19135]
NOTE In the case of an IHO Register, the Register Manager performs the functions of the registration authority specified in the IHO Directives.

register owner
organization that establishes a register
[ISO 19135]

registration
assignment of a permanent, unique (in the register), and unambiguous identifier to an item
[ISO 19135]

registry
information system on which a register is maintained
[ISO 19135]

relationship
semantic connection among model elements [ISO 19103]
NOTE Kinds of relationships include association, generalization, metarelationship, flow, and several kinds grouped under dependency.

remote sensing
collection and interpretation of information about an object without being in physical contact with the object

render
conversion of digital graphics data into visual form
[ISO 19117]
EXAMPLE Generation of an image on a video display.

resolution (of a sensor)
smallest difference between indications of a sensor that can be meaningfully distinguished
NOTE For imagery, resolution refers to radiometric, spectral, spatial and temporal resolutions.
[ISO/TS 19101-2:2008]

resource
asset or means that fulfils a requirement
EXAMPLE Dataset, service, document, person or organisation.
[ISO 19115:2005]

retirement
declaration that a register item is no longer suitable for use in the production of new data [ISO 19135]
NOTE The status of the retired item changes from ‘valid’ to ‘retired’. A retired item is kept in the register to support the interpretation of data produced before its retirement.

ring
simple curve which is a cycle [ISO 19107]
NOTE Rings are used to describe boundary components of surfaces in 2-D coordinate systems.

schema
formal description of a model
[ISO 19101]

scripting catalogue
generic term describing a collection of one or more files containing scripting functions

scripting domain
the application of scripting to an S-100 domain, such as portrayal

scripting engine
a Lua interpreter or virtual machine

scripting function
a function written in Lua

semi-major axis
semi-diameter of the longest axis of an ellipsoid
[ISO 19111]

semi-minor axis
semi-diameter of the shortest axis of an ellipsoid
[ISO 19111]

sensor
element of a measuring instrument or measuring chain that is directly affected by the measurand
[International Vocabulary of Basic and General Terms in Metrology (VIM)]

sensor model
description of the radiometric and geometric characteristics of a sensor
[ISO19101-2:2008]

server
a technical entity (for example: Device, Program) that offers a service to a client

service specification
the purpose of a service specification is to provide a holistic overview of one particular service and its building blocks at logical level including the A-Profile. It may be complemented by a model based description (for example, UML model describing the service interfaces, operations and data structures). The service specification describes a well-defined baseline of the service and clearly identifies the service version

session
set of client service communication. A session is set up or established at a certain point in time, and then torn down at some later point. An established communication session may involve more than one message in each direction. A session is stateful, meaning that at least one of the communicating parts needs to save information about the session history in order to be able to communicate, as opposed to stateless communication, where the communication consists of independent requests with responses

spatial reference
description of position in the real world

spatiotemporal domain <coverage>
domain composed of geometric objects described in terms of spatial and/or temporal coordinates
[ISO 19123]
NOTE The spatiotemporal domain of a continuous coverage consists of a set of direct positions defined in relation to a collection of geometric objects.

specification
declarative description of what something is or does
NOTE Contrast: implementation.

specification scope
a partitioning of the data content of the product on the basis of one or more criteria
[adapted from ISO 19131]

spectral resolution
specific wavelength interval within the electromagnetic spectrum
EXAMPLE Band 1 of Landsat TM lies between 0.45 and 0.52 µm in the visible part of the spectrum.

standard catalogue functions
scripting functions which are guaranteed to be part of all scripting catalogues

standard host functions
scripting functions which must be provided by the host. Scripting functions call standard host functions to obtain information about the dataset(s) being processed

standard scripting functions
All scripting functions defined within S-100 Part 13. The union of Standard Host Functions and Standard Catalogue Functions

start point
first point of a curve
[ISO 19107]

stereotype
new type of modelling element that extends the semantics of the metamodel [ISO 19103]
NOTE Stereotypes must be based on certain existing types or classes in the metamodel. Stereotypes may extend the semantics, but not the structure of pre-existing types and classes. Certain stereotypes are predefined in the UML, others may be user defined. Stereotypes are one of three extensibility mechanisms in UML. The others are constraint and tagged value.

stream
in online data exchange: a continuous sequence of fragmented data to be transported by a communication system

sub-attribute
an attribute of a complex attribute

subfield
a subfield is a component of a *field*. It is a contiguous string of bytes whose position, length and data type are described in the field data description. It is the smallest unit of information which can be described by this standard
NOTE Certain stylized subfields, such as date (YYYYMMDD), must be further resolved by an application.

submitting organization
organization authorised by a register owner to propose changes to the content of a register
[ISO 19135]

subregion
collection of geospatial data for a specific area within a dataset where the geospatial data conforms to a common, specific acquisition requirement that may differ from that of other collections within the cell

supersession
declaration that a register item has been retired and replaced by one or more new items [ISO 19135]
NOTE The status of the replaced item changes from 'valid' to 'superseded.' A superseded item is kept in the register to support the interpretation of data produced before its supersession.

surface
connected 2-dimensional geometric primitive, representing the continuous image of a region of a plane [ISO 19107]
NOTE The boundary of a surface is the set of oriented, closed curves that delineate the limits of the surface.

surface patch
2-dimensional, connected geometric object used to represent a continuous portion of a surface using homogeneous interpolation and definition methods
[ISO 19107]

symbol
portrayal primitive such as linestyles, patterns, text and point symbol graphics defined in SVG

tag
an ISO/IEC 8211 implementation concept used to identify each instance of a *field*

tag value
explicit definition of a property as a name-value pair

NOTE In a tagged value, the name is referred as the tag. Certain tags are predefined in the UML; others may be user defined. Tagged values are one of three extensibility mechanisms in UML. The others are constraint and stereotype.

technical service
taken from the concepts of service-oriented architectures, a technical service refers to a set of related software functionalities that can be reused for different purposes together with policies that govern and control its usage. A technical service is a service offered by an electronic device to another electronic device. Often operational services are implemented by electronic devices that offer several technical services to use the operational service

temporal reference system
reference system against which time is measured

tessellation
partitioning of a space into a set of conterminous geometric objects having the same dimension as the space being partitioned [ISO 19123]
NOTE A tessellation composed of congruent regular polygons or polyhedra is a regular tessellation; One composed of regular, but non-congruent polygons or polyhedra is semi-regular. Otherwise the tessellation is irregular.

triangulated irregular network (TIN)
tessellation composed of triangles
[ISO 19123]

tuple
ordered list of values

type
stereotype of class that is used to specify a domain of instances (objects) together with the operations applicable to the objects
NOTE A type may have attributes and associations.

unit
defined quantity in which dimensioned parameters are expressed

value
element of a type domain [ISO/TS 19103:2005]
NOTE 1 A value may be considered a possible state of an object within a class or type (domain).
NOTE 2 A data value is an instance of a data type, a value without identity.

value domain
set of accepted values [ISO/TS 19103:2005]
EXAMPLE The range 3-28, all integers, any ASCII character, enumeration of all accepted values (green, blue, white).

vertical coordinate reference system
one-dimensional coordinate reference system based on a vertical datum
[ISO 19111]

vertical coordinate system
one-dimensional coordinate system used for gravity-related height or depth measurements
[ISO 19111]

vertical datum
datum describing the relation of gravity-related heights or depths to the Earth
[ISO 19111]