Motivation for a GeoJSON Encoding Schema

Submitted by B.R.Calder & G.Zelenak

SUMMARY

Executive Summary: This document provides a rationale for the proposal, originally made at CSBWG13, to acknowledge, and potentially endorse, a community-led project to provide a GeoJSON metadata schema for B.12 data.

Action to be taken: Review and discuss at CSBWG14

Related documents:None

[Note: an open source project to support a JSON schema for the B.12 metadata already exists (https://github.com/CCOMJHC/csbschema), but for the purposes of this document, the argument is made as if it did not. The purpose here is to demonstrate why the extant project already exists, and is developing quickly: it satisfies a real world, current need; and to make the case for interaction between this community project and the CSBWG.]

Introduction

The metadata requirements for a B.12-compliant data file are specified in B.12 as mandatory, recommended, and extended. However, although the metadata requirements can be extensive, B.12 defines only the content of the metadata, and does not specify how it should be represented in a file to be sent to the DCDB for archive. It is therefore possible that multiple different (B.12-compliant) variants of the metadata could be constructed and submitted, and the archive be left to work out the details. This is likely to cause confusion, difficulties in normalising the database, and interpretation ambiguities for users extracting data from the DCDB. This is an unfortunate situation.

The current solution to this dilemma is to provide a single example JSON-encoded file from the DCDB website. Although this shows the broad strokes of a plausible encoding, it cannot show all potential components of a complete metadata record, and more importantly has no mechanism to indicate which text fields are free-form, and which have controlled vocabulary, etc. While helpful, this is not a complete solution, and cannot readily be validated or extended. Another solution is required.

Potential Solutions

In most examples, a metadata (or other data format) specification is accompanied by an encoding guide, or (better) a product specification. These documents, ideally machine as well as human readable, specify uniquely how the content of the metadata specification should be rendered in a particular data format. In this instance, since the DCDB mandates that metadata for the CSB database be encoded in JSON (JavaScript Object Notation) format so that it can be combined with the data in GeoJSON format, the target has to be a JSON rendering of B.12. Note that an encoding of the metadata describes only the structure of the metadata, and not the content. Therefore any personal data that might potentially be in the metadata would not be part of the schema.

A number of different potential solutions are possible. First, the IHO, through the CSBWG or otherwise, could commission an effort to write and publish an encoding guide companion document to B.12 explaining in detail how to translate the metadata recommendations into JSON format.

Second, the CSBWG, as currently constituted, could commission a sub-group to develop a companion document to B.12 that would provide recommendations for encoding to JSON.

Third, the community (outside, although potentially associated with, CSBWG) could develop a guide to encoding, ideally in a form that is machine-readable.

Analysis of Solutions

Although the IHO solution would have the significant benefit of the weight of an official standard, the time requirement to commission the effort, and then to complete it, is likely order 18-24 months, plus potentially another 12 months for ratification. In the meantime, variant JSON-encoded metadata files are being generated, and potentially polluting the DCDB. This is therefore not recommended.

The CSBWG solution suffers from the same issues. Although it would not take as long to commission the document, the development time would likely be the same, and the result would likely also require a formal review (and potentially a circular letter), delaying the adoption for a considerable period. As with the IHO solution, this would result in potentially many variants of JSON-encoded metadata being deposited in the DCDB in the meantime.

The community-led effort of the third proposed solution is the least formal, and will have at most the status of a *de facto* standard. However, since it can proceed immediately, and does not require a formal adoption (i.e., a Trusted Node can choose to use the resulting information or not at their discretion), it is likely to be significantly faster to first use, and therefore has the potential to reduce the amount of variant JSON metadata that enters the DCDB. The lack of formal status could potentially be ameliorated by the CSBWG acknowledging such a project, and making a strong recommendation that Trusted Nodes consider using the project's outputs to structure their metadata.

It is therefore recommended the the CSBWG support the development of a community-led metadata encoding.

A Community-led Metadata Encoding

Traditionally, an encoding guide would be a structured text document that describes each potential element of the metadata and the values that can be used. Such guides are often extensive, verbose, and difficult to use (specifically, they are usually not machine-readable and therefore need to be translated - potentially with translation errors - into executable form).

As an alternative, JSON has a schema mechanism where a specially constructed JSON file specifies, in standard format, the structure that is expected in a target JSON file. Constructed appropriately, the schema file can be used to validate any given JSON file to ensure that it meets the appropriate requirements. For example, if a particular tag in the JSON metadata is only allowed to contain a particular set of values (usually termed a "controlled vocabulary"), the options can be specified in the schema and the validation code can check that the target file does not contain anything not on the controlled list. A specific example might be the vertical depth reference, which should be defined as "Transducer" (i.e., depth from the echo sounder), "Waterline" (i.e., depth corrected to the current ship's draft), or "Unknown" (i.e., anything else), but no other value.

Similarly, if a tag in the metadata must contain certain information, the schema can specify which information is required. For example, the B.12 guidelines mandate specific metadata be present in the

platform description; the schema can be used to ensure that this is the case. Alternatively, if only one of a limited subset of tags is allowed, the schema can list all of the options, and indicate that only one can be selected. Violations of these constraints are automatically detected through use of the automated validation software.

There are some things that cannot be expressed in the JSON schema format. Specifically, context-dependent checks (i.e., where the value of one tag deepens on the value of another) cannot be encoded, and have to be enforced separately. A specific example in the CSB metadata is the ID number (e.g., an IMO identifier or MMSI) where the format cannot be specified unless the ID type tag is examined first. These can be enforced with additional code.

Developing a JSON schema is relatively simple, but the mechanisms surrounding the development can be more involved if the process is going to be effective. This is a standard problem in the open source community, and therefore tools have been developed to support and foster cooperation. Specifically, distributed source-code control (with, e.g., git) to maintain a coherent view of the source code with spatially distributed developers; a hosted repository (e.g., through GitHub) to make the code available to all of the developers and potentially (as read-only) the public; issue trackers (e.g., GitHub Issues, or JIRA) to provide a controlled approach to development requests, bug reports, etc.; and wikis (e.g., GitHub Pages or ReadTheDocs) to provide documentation and communication on the development effort are all generally available. A project of this kind would typically be hosted in a public repository space, such as GitHub, and managed by a lead organisation on behalf of the project; the details (e.g., who maintains the repository) can be adjusted as required by the project.

These tools, however, do not form the community: they just support it. In practice, unless a community develops around the project, it is unlikely to garner enough developer support to grow and mature, or to be adopted in practice across the community (which is a prerequisite for practical utility). In the CSB case, however, there is already a community of developers with a vested interest in rapidly resolving the metadata encoding problem, and therefore it is unlikely to be difficult to pull together an initial developer community to support the schema construction and maintenance.

Therefore, it appears to be the case that there is motivation, interest, and technical facilities to support the development of a community-led schema-based metadata encoding guide within the CSB community.

Coordination and Governance

Evidence in the last twelve months has demonstrated that rapid development in CSB hardware, software, and metadata is taking place. Particularly, this development pace is significantly faster than that which can be supported by the typical meeting tempo of an IHO working group, even with intercessional work. An independent, community-driven schema project fosters this form of development, at a pace matched to real-world requirements.

It can also, however, pose questions of coordination with the B.12 document. That is, if the schema project develops to match community needs, it must necessarily extend or modify the definition of metadata in the current version of B.12; although this is a necessary part of the development process, without careful coordination it could lead to a *de facto* definition of what the GeoJSON metadata should look like that is at odds with what the working group would recognise.

The methods used for most development projects (i.e., Git, GitHub, and an issue tracker) provide for a communications channel to enable this coordination requirement. Specifically, since the B.12 maintenance effort is using an issue tracker, any recommendations for changes to metadata can be made in the schema repository using conventional development techniques, and then raised as an issue in the B.12 maintenance repository for integration. The issue tracking mechanism allows for transparency of process, tracked comments (from anyone with an interest in the topic) for discussion,

and tight integration with the development of B.12 in the associated repository. Appropriate tagging in the schema repository would allow the specific recommended changes to be fully identified. In practice, the schema repository would be used to develop and test new, extended, or modified encoding rules as required, which would be considered "experimental" until they became a formal proposal to the B.12 maintenance group. After formal adoption through the B.12 maintenance group, the "experimental" changes would be formalised as a new version of the metadata encoding.

The issue tracking mechanisms, and requests for update to the repository (known as "pull requests") also address the issues of open discussion of proposed modifications, documentation of reasoning for changes, and a mechanism to revert those changes if mistaken. Normal processes for release scheduling, such as providing an implementation of the proposed changes as a separate branch of the repository (a "release candidate") with time for public comments, also address concerns about review, shared governance, and informed consent.

These mechanisms may need some initial development, but prove sufficient for a large variety of very complex development projects (including, for example, the entire Linux system, for which Git was designed). There are very likely, therefore, sufficient for a relatively small, constrained development process such as a GeoJSON schema.

Recommendations

1. The CSBWG is invited to encourage the development of a community-led JSON schema to validate the JSON metadata specified in the B.12 guidance document.
2. The DCDB is invited to actively engage with the community-led development effort in order to ensure that the schema developed meets their requirements for archival.
3. On development, the CSBWG is invited to acknowledge version 3.0.0 (or subsequent version) of the schema and associated validation software, consistent with B-12 Edition 3.0.0 (or subsequent amended version) and corresponding sample formats hosted by the DCDB and previously reviewed by the working group. Further, the CSBWG is invited to officially recommend that Trusted Nodes utilise version 3.0.0 (or subsequent version) of the schema as an encoding guide, and the associated software to validate the constructed metadata on each file before submitting it to the DCDB for archive.
4. The CSBWG is invited to request the B.12 maintenance sub-group to monitor and/or participate in the development of the community-led project, and to coordinate updates to B.12 to reflect community-driven modifications to the schema.
5. The CSBWG is invited to consider future effort to officially adopt the community-led project, after development, as a model for a formal standard for encoding.