OGC® DOCUMENT: 24-999

External identifier of this OGC® document: http://www.opengis.net/doc/is/poi/1.0

Open Geospatial Consortium

OGC POINTS OF INTEREST (POI) INTERNATIONALIZED TEXT ANALYSIS

TECHNICAL PAPER

DRAFT

Version: 1.1
Submission Date: 2024-11-16
Approval Date: 2024-11-16
Publication Date: 2024-11-16
Editor: Charles Heazel, Howard Trickey

Note

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

I ABSTRACT

The OGC Points of Interest (POI) Conceptual Model is an open data model for representing information about POI. The model is defined using a Unified Modeling Language (UML) object model. This UML model extends the ISO Technical Committee 211 (TC211) conceptual model standards for spatial and temporal data. Building on the ISO foundation assures that the features described in the POI Model share the same spatiotemporal universe as described by related standards (e.g., CityGML).

The goal for developing the OGC POI Conceptual Model is to reach a common definition of the basic entities, attributes, and relations of “points of interest.” In the broadest terms, a POI is a location about which information of general interest is available. A POI can be as simple as a set of coordinates and an identifier, or more complex such as a three-dimensional model of a building with names in various languages, information about open and closed hours, and a civic address.

II KEYWORDS

The following are keywords to be used by search engines and document catalogues.

ogcdoc, OGC document, Internationalization

III PREFACE

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

IV SECURITY CONSIDERATIONS

The POI Conceptual Model defines a POI as a type of Feature. By building on the same Feature Model as other OGC Feature models, POI implementations inherit the security controls and vulnerabilities of their associated Feature Dataset. They are a Feature like any other.

This document defines a Conceptual Model Standard. Implementations of this Standard (Implementation Specification) are free to add additional details and content necessary to enable implementation-specific security controls. In the event that anything in this Standard prevents implementation of needed controls, implementors are requested to notify the POI Standards Working Group (SWG) and help devise a solution.

V SUBMITTING ORGANIZATIONS

The following organizations submitted this Document to the Open Geospatial Consortium (OGC):

- Digital Flanders
- Google
- HeazelTech
- Pangaea Innovations
- PEREY Research and Consulting
- US Army Geospatial Center

A

ANNEX A (INFORMATIVE) INTERNATIONALIZED TEXT

A

ANNEX A (INFORMATIVE) INTERNATIONALIZED TEXT

A POI is a location about which information of general interest is available. A POI can be as simple as a set of coordinates and an identifier, or more complex such as a three-dimensional model of a building with names in various languages, information about open and closed hours, and a civic address. It follows that support for language-specific names, or internationalized text in general, is a reqirement for a successful POI Standard.

The POI Standard builds on the Geospatial Feature and Geometry Standards developed by the Open Geospatial Consortium and ISO Technical Committee 211 (TC211). It should not be necessary to develop an approach for internationalized text which is specific to POIs. That support should already exist within the foundational standards. This appendex seeks to provide a window into that support. To make explicit the support for internationalized text which is implicit in the international standards.

A.1. International Standards

This section discusses how the Standards developed by ISO TC211 and the OGC support internationalized text.

A.1.1. Conceptual Schema

ISO 19103 defines the conceptual schema language (CSL) for developing computer interpretable models, or schemas, of geographic information. This includes definition of the primitive data types used in OGC and ISO TC211 developed standards. CharacterString is one of those primitive data types.

ISO/TS 19103:2005 — Geographic information — Conceptual schema language was the first version of ISO 19103. This version defined the CharacterString class as a primitive type with attributes for CharacterSetCode, maxLength, size, and elements (characters).

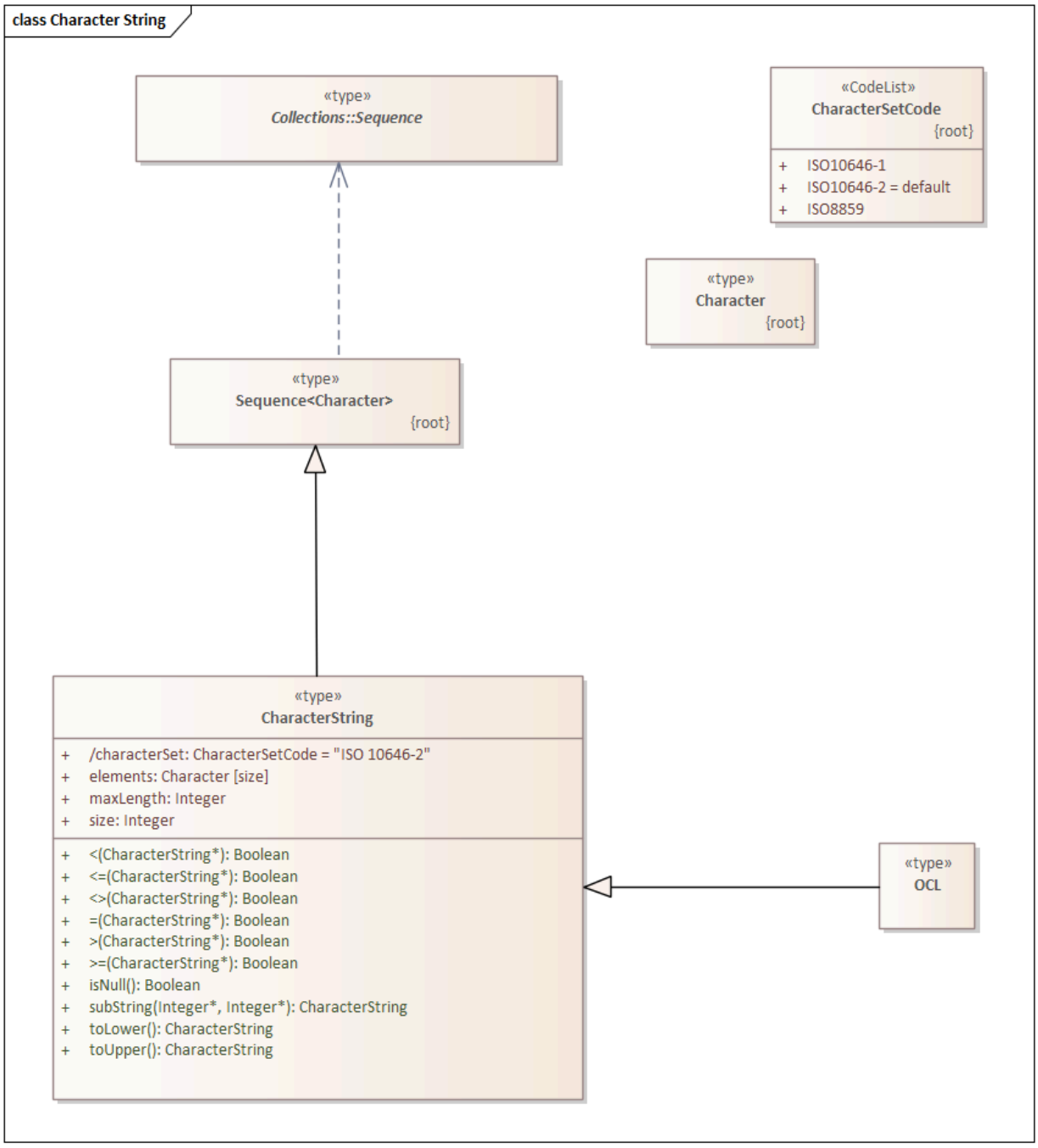

Figure A.1 — CharacterString Context.

ISO 19103 was updated by ISO 19103:2015 — Geographic information — Conceptual schema language.

In this version, the CharacterString class has the same information content as in 19103:2005. However, this class is modeled as an interface instead of a type. It also adds a normative Annex C "Data types – extension types".

Section C.2 of this annex addresses "Cultural and Linguistic Adaptability" which includes a new "LanguageString" class. LanguageString is a subclass of CharacterString. This subclass

adds a languageCode whose values come from ISO 639. As a result, a LanguageString is a CharacterString with a associated language code.

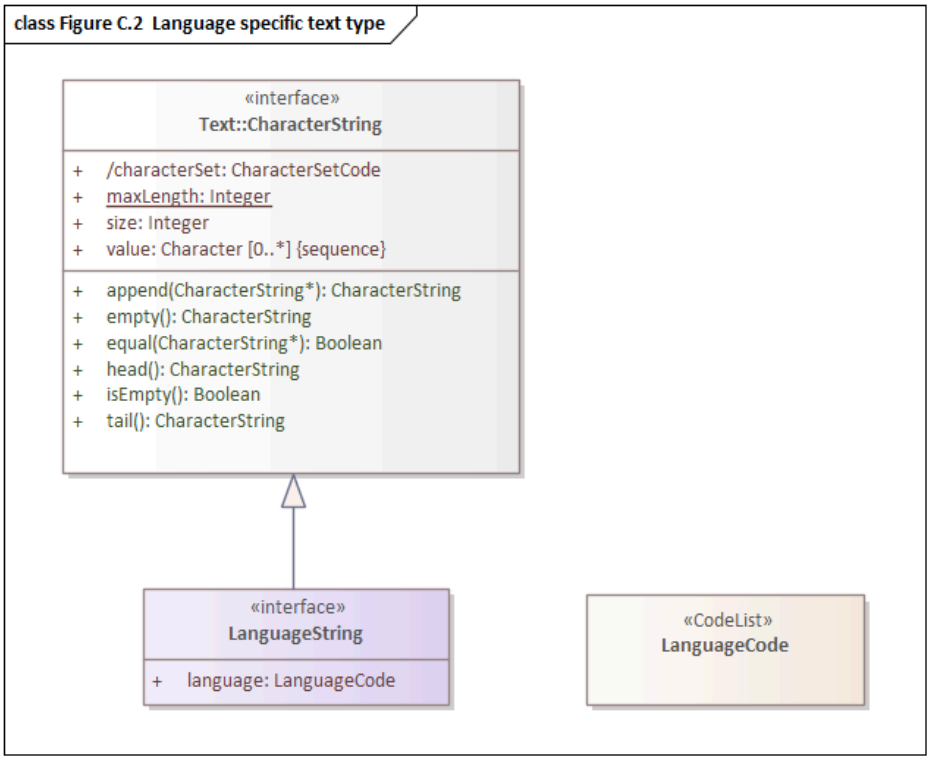

Figure A.2 — LanguageString Context.

A.1.2. Location-Based Services

The Location-Based Services family of standards was developed to extend Web-based location services to the bandwith limited cell phones of the time. This is arguably the first appearance of a language-specific text field in an ISO TC211 Standard.

ISO 19132:2007 “Location-based Services Reference Model” introduces the LanguageSpecificCharacterString class. This is a subclass of the CharacterString class from 19103:2005. Its sole function is to a LanguageCode attribute to CharacterString.

Since LanguageSpecificCharacterString is a subclass of CharacterString, any CharacterString can (in principle) be a LanguageSpecificCharacterString.

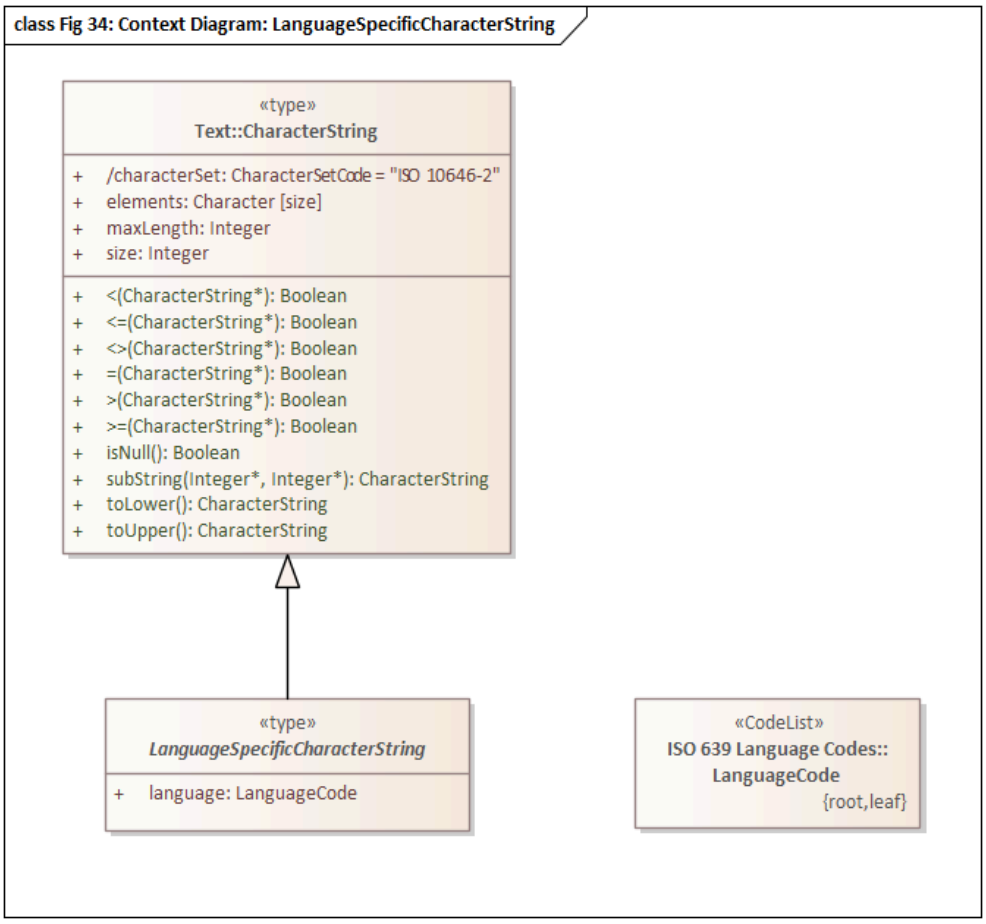

Figure A.3 — LanguageSpecificCharacterString Context.

A.1.3. Metadata

There are two series of TC211 Standards for Metadata. The ISO 19115 series provides the conceptual models. The ISO 19139 series defines XML schema for the ISO 19115 Standards.

A.1.3.1. ISO 19139

The ISO 19139:2007 “Metadata XML Implementation” Standard defines the XML schema for ISO 19115:2003. ISO 19139 also addresses internationalization, but it uses a different approach from that used in ISO 19132.

ISO 19139 introduces the LocalizedCharacterString class. This class is a realization of the CharacterString class from ISO 19103:2005. LocalizedCharacterString adds to CharacterString an association with the new PT_Locale class. PT_Locale includes attributes for CharacterSetCode, Country, and LanguageCode. Only the LanguageCode is required.

Note that CharacterSetCode is already defined by CharacterString so it is redundant in PT_Locale.

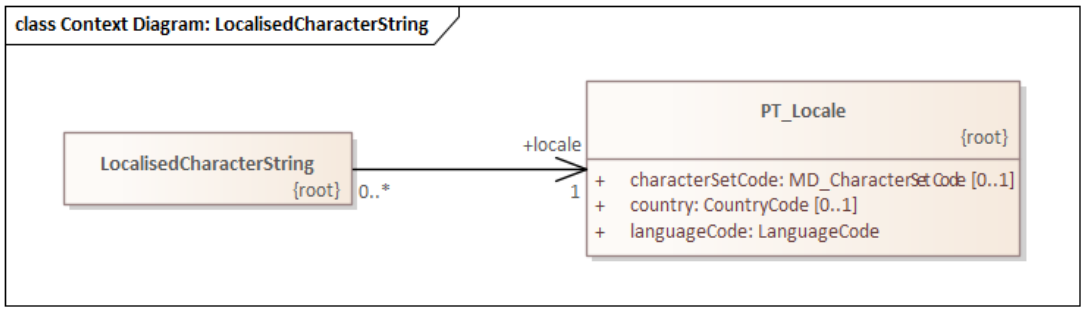

Figure A.4 — PT_Locale Context.

A.1.3.2. ISO 19115

ISO 19139:2007 defines the XML schema for ISO 19115:2003. However, the PT_Locale class does not appear in ISO 19115:2003. So initially PT_Locale was only defined for XML metadata encodings.

This limitiation was partially lifted by ISO 19115-1:2014. This update to ISO 19115 moves PT_Locale up to the conceptual level. It is no longer limited to XML encodings. The XML encoding standard for 19115-1 is ISO 19115-3:2016. This Standard continues the support the PT_Locale from ISO 19139:2007.

At this time XML is the only standardized encoding for ISO 19115. Unitil other encodings are defined, PT_Local will continue to be a concept for use in XML metadata encodings.

A.1.4. OGC Web APIs

"OGC API — Common — Part 1: Core" provides a brief discussion of text internationalization in section 8.6. It includes a recommendation thbat "For encodings that support string internationalization, the server SHOULD include information about the language for each string value that includes linguistic text."

The text internationalization techniques it describes are:

1. Use of the HTTP `Accept-Language` header to convey the language desired by the requestor. This approach assumes that the server has multiple copies of the resource, each in a different language, or that the server is capable of generating a language-specific version based on the `Accept-Language` header value.

2. Use of the `hreflang` attribute of the `Link` class. The `Link` class is based on RFC 8288 (Web Linking). It provides the information needed to properly invoke an HTTP request. This includes the URL, format, and language of the target resource.

3. The built-in capabilities of JSON-LD.

The same content appears in "OGC API — Features — Part 1: Core" Section 7.10.

Since most OGC API standards build off of API-Common or API-Features, they all inherit the same recommended approaches.

A.1.5. JSON-LD

One of the capabilities that JSON-LD adds to JSON is the ability to annotate strings with their language. The capabilities provided include:

language-tagged string type: A language-tagged string type consists of a a string and a non-empty language tag. This is implemented using the rdf:langString type.

`@language` keyword: The `@language` keyword is used to specify the language for a particular string value or the default language for use within a JSON-LD document. When used within a `context`, it specifies the default language to be used within the scope of that context. If used within the top-level context, it defined the default language for the whole document unless overridded by a lower level context.

Text internationalization is addressed in section 4.2.4 of the JSON-LD 1.1 Standard (https://www.w3.org/TR/json-ld/#string-internationalization)

Language Maps: Allows multiple values differing in their associated language to be indexed by language tag.

Language Maps are addressed in section 9.8 of the JSON-LD 1.1 Standard (https://www.w3.org/TR/json-ld/#language-maps)

A.1.6. Conclusion

All standards which build on the ISO 19103:2015 Standard have the option to use the “LanguageString” class for text data. This class allows the association of a language identifier with the text string. In some encoding languages, addition of this attribute to a CharacterString is easily accomplished. Others may require definition of a LanguageString class to support the additional attribute.

The PT_Locale concept is useful, particularly if the locality information includes more than just the language code. Use of PT_Local establishes a dependency on the ISO 19115 metadata standards. However, there are many additional data types defined in these Standards which deserve reuse. Such a dependency may be a good practice.

JSON-LD provides powerful text internationalization capablities, which unfortulately are only applicable to that encoding language.

A.2. Common Practice

Three general techniques for internationalized text are found in common practice:

Table A.1

Implied Language	No language specified with each text string. Perhaps there is a file-scoped language, known by some external means or by one file-scoped metadata element.
Separate Per-Language Attributes	Multiple copies of the same text element that have the language as a suffix (e.g., “name”, “name:fr”, “name:en”, ...) with the unsuffixed version representing a default langauge.
Complex Attributes	In this case text strings are sequences or objects, not character strings. Each element of the sequence is an object has a character string (the text itself) along with an attribute that says what language it is in. If the name is an object, then it has a default-language string and optional other-language strings in a sequence as described previously.

A.2.1. Implied Language

Implied Language is the most common approach to internationalized text. When using this technique, all of the text in a document, or section of a document, is in the same language. Support for multiple languages requires duplication of the entire section for each language.

A.2.2. Per-Language Attribute

The per-language attribute approach tags the text string with a language identifer. This identifier can either be a part of the string itself, or an attribute associated with the text string element.

For example, if we wanted to provide the name of the Statue of Liberty in XML using both French and English, then we could use:

A language identifier embedded in the element value

```
<name>"Statue of Liberty:en"</name>
<name>"Statue de la Liberté:fr"</name>
```

Figure A.5

Or provide the language identifier as an attribute of the element

```
<name language="en">"Statue of Liberty"</name>
<name language="fr">"Statue de la Liberté"</name>
```

Figure A.6

A.2.3. Complex Attributes

There are several variants of the Complex Attribute method:

A.2.3.1. Complex Attributes: Sequence-only

Every text string must be a sequence of objects. It might look something like this:

```
"name": [
  {"name" : "Statue of Liberty", "language" : "en"},
  {"name" : "Statue de la Liberté", "language" : "fr"}
]
```

Figure A.7

A.2.3.2. Complex Attributes: Default and Sequence

Here the text string is an object that contains a default name and a sequence. The language of the default name could be specified by a “language” property in the outermost object of the file.

It would look something like this:

```
"name" : {
  "default" : "Statue of Liberty",
  "names": [
    {"name" : "Statue of Liberty", "language" : "en"},
    {"name" : "Statue de la Liberté", "language" : "fr"}
  ]
}
```

Figure A.8

A.2.3.3. Complex Attributes: Inspire

To follow existing standards, the Inspire method could be used. The Inspire method is based on the PT_Locale concept described above.

This approach is very similar to the Sequence-only method but the elements of the sequence, instead of referring to *languages*, refer to *locales*. This requires definitions of a number of **locales** somewhere at the file scope.

Adapted to JSON, it might look something like this:

```
"locales" : [
  {
     "id" : "locale_en",
     "language_code" : {
        "code_list" : "link:++http://www.loc.gov/standards/iso639-2/"++[],
        "code_list_value" : "en",
        "name" : "English"
     },
     "character_encoding" : {
        "code_list" : "resources/codelist/gmxcodelists.xml#MD_
CharacterSetCode",
        "code_list_value" : "utf8">UTF 8</MD_CharacterSetCode>,
        "name" : "UTF8"
     }
```

```
      },
      {
         "id" : "locale_fr",
         "language_code" : {
            "code_list" : "link:++http://www.loc.gov/standards/iso639-2/"++[],
            "code_list_value" : "fr",
            "name" : "French"
         },
         "character_encoding" : {
            "code_list" : "resources/codelist/gmxcodelists.xml#MD_
CharacterSetCode",
            "code_list_value" : "utf8">UTF 8</MD_CharacterSetCode>,
            "name" : "UTF8"
         }
      }
    ]
```

Figure A.9

Then an actual name attribute would look something like:

```
    "name": [
      {"name" : "Statue of Liberty", "locale" : "locale_en"},
      {"name" : "Statue de la Liberté", "language" : "locale_fr"}
    ]
```

Figure A.10

A.2.3.4. Complex Attributes: Choice of Simple or Complex

In this variant, the text string can be one of two types: a simple string or one of the other Complex Attribute variants. If the value is only a simple string, then it is assumed to be in the default language, specified at file scope. Otherwise, the structured value will give all of the desired language variants.

The advantage of this variant is that the simple case of all-one-language yields a file that is simple to understand and process.

A.3. Discussion and Recommendation

Here is a table of some pros and cons of the various methods discussed.

Table A.2

METHOD	PROS	CONS
Implied Language	Simple. Easy to write and use.	The only way to handle multiple languages is to provide a choice of multiple files. Keeping such files in sync is error-prone.

METHOD	PROS	CONS
Separate Per-Language Attributes	Familiar to OpenStreetMap users. Handles the one-language case well. Compact.	Conceptually unclean to have a number of name attributes at the same level as other attributes. Needs more post-processing to gather together all the names in the internal data format.
Complex Attributes: Sequence Only	Moderately simple to read and process.	Bulkier for one-language case. Not clear what the default name to use is.
Complex Attributes: Default and Sequence	Handles one-language case moderately well. Can tell what name to use by default.	Bulkier than most other options, and still not ideal for one-language case.
Complex Attributes: Inspire	Closest to “standards compliant”. Allows specification of character encoding too, and in a less-verbose way than if done per name.	Complex to read and write (the locales part). Need to process metadata in another part of the file and connect to each POI: a POI feature would not be standalone. Extra complexity of indirection is only useful if multiple character encodings are needed.
Complex Attributes: Choice of Simple or Complex	Handles one-language case well. Any other pros of the variant of Complex that is coupled with this.	Need to make value-type-dependent choice when processing.

The recommendation is to use the Choice of Simple or Complex attributes, and in the case of a Complex attribute, use the Sequence Only submethod.