S-100 Maintenance - Change Proposal Form

TSMAD27-4.3.10A

Organisation	Jeppesen	Date	2013-11-01
Contact	Eivind.Mong@jeppesen.com	Email	Eivind.Mong@jeppesen.com

Change Proposal Type *Select only one option*

1.Clarification	2.Correction	3.Extension
		Y

Location *Identify all change proposal locations*

Note: (new) means numbering as in the accompanying redline document.

S-100 Version No.	Part No.	Section No.	Proposal Summary
1.0.0	1	3	Add the standards for dictionaries/thesauri to the references
	1	4.7	Insert new clause 4.7 on CodeList types
	1	4.8.1 (4.10.1 in TSMAD26 redline of edition 2.0.0)	Add item (f) describing UML model of codelists
	2a	4.2	Add "codelist" to the attribute data types
	3	5.2.8	Add clarification to **S100_GF_AttributeType** for domains of codelist type
	3	5.3	Add attribute type for codelists
	5-A	Figure A-1	Add codelist to the enumeration of attribute types
	New Annex 11a (Informative)	New Annex	Add the Guidelines for Product Specification Authors annex in the accompanying paper as an informative annex

Change Proposal

1-3 Normative References: Add
ISO 19136: Geographic Information – Geography Markup Language
OGC 10-129r1: Geographic Information – Geography Markup Language (GML) – Extended schemas and encoding rules
SKOS: SKOS – Simple Knowledge Organization System – Reference. W3C Recommendation, 2009. http://www.w3.org/TR/2009/REC-skos-reference-20090818/.
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

(Insert) **1-4.7 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 (e.g., distribution media).

A codelist type declaration defines either:

- a list of valid key-value combinations (i.e., code-value mappings) with a provision for allowing user communities to provide allowed values in a specified format; or,
- 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.

Code lists are modelled as classes that are stereotyped as <<Codelist>>. Code lists of the first form must list the known literals as attributes. In the second form, no attributes are listed. A CodeList classifier must have tagged values which define its representation and extensibility, and may have a tagged value which hints at the anticipated encoding. **Figure 1** shows two examples of codelists – the Languages codelist is an example of a codelist modelled as an extensible enumeration (indicated by the tagged values *asDictionary=false* and *extensible=true*) and the Countries codelist is an example of a codelist modelled by an external dictionary (indicated by tagged value *asDictionary=true*) whose location is given by its *vocabulary* tagged value.

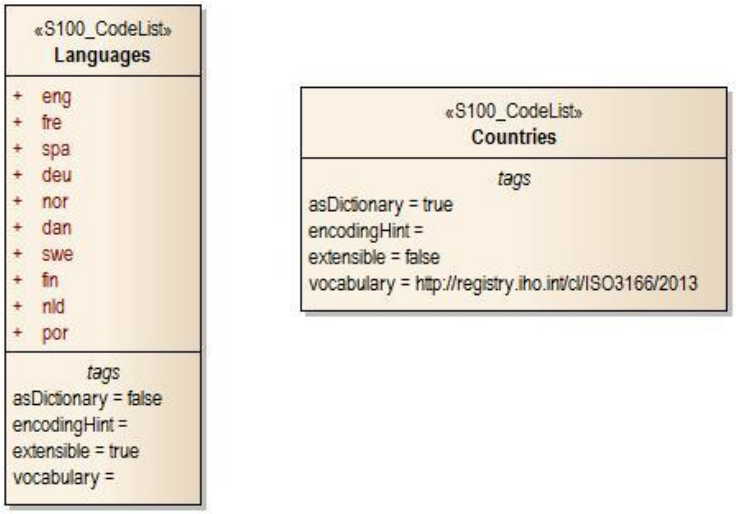

Figure 1. Codelists

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.

The tagged values for S100 CodeLists are described in the table below.

Table 1. Tagged values for codelists

Model	Tag			
	asDictionary	vocabulary	extensible	encodingHint

enumeration with pattern	false	(nil)	true: additional values permitted (default) false: additional values not permitted	enum: encode as ordinary enumeration (must have extensible=false) open: encode as union of list and pattern "other: \w{2,}" (default) + others as defined in product specifications
dictionary	true	(URI)	true: additional values permitted false: additional values not permitted (default)	enum: encode as ordinary enumeration (must have extensible=false) resource: encode as URI pointing to item in vocabulary (default) open: encode as URI identifying an item in either the specified vocabulary or another vocabulary + others as defined in product specification, or empty

1-4.8.1 (1-4.10.1 in TSMAD 26 draft of Edition 2.0.0) Use of standard UML stereotypes for class/classifier

(Add) 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 (e.g., 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).

2a-4.2: Add "codelist" to the attribute data types.

3-5.2.8 S100_GF_AttributeType: add sentence explaining that for a codelist the domain of values may be given by a URI identifying a "vocabulary".

3-5.3: Add S100_CodeListAttributeType to the UML diagram in Figure 3-2 – Attributes.

5-A Figure A-1: Add codelist to the enumeration in S100_CD_AttributeDataType

New **Annex 11c (Informative) – Guidance on Codelists:** Add Annex A of the accompanying paper "Codelists" as an informative annex.

Change Proposal Justification

Please provide a suitable explanation for the change and where applicable supporting documentation.

Codelists are a method of representing open, flexible enumerations. They are suitable for modelling lists of values not all of which are knowable when the application schema is developed, or when the list may need to be extended without a major revision of product specifications. They are also suitable for long lists of values which are shared by multiple product specifications and/or maintained by an external authority.

An accompanying TSMAD 27 paper contains details on the proposal.