INTERNATIONAL HYDROGRAPHIC ORGANIZATION

IHO GUIDELINES FOR CREATING S-100 PRODUCT SPECIFICATIONS

IHO Publication S-97

Edition 1.0.0 – Xxxx 2020

PART B

EXECUTION

Document History

Changes to this Specification are coordinated by the IHO S-100 Working Group. New editions will be made available via the IHO web site. Maintenance of the Specification shall conform to IHO Resolution 2/2007 (as revised).

Version Number	Date	Author	Purpose
0.1	31 Dec 2017	RM, EM	First draft
0.2	31 Aug 2018	RM, EM	Addressed comments from NIPWG and S-100WG; updates for conformity to S-100 Ed. 4.0.0; new material for services; new material for metadata; new material for data protection, authentication, and compression; miscellaneous editorial updates.
1.0.0 RC1	11 Mar 2019	RM; JW	IHO Secretariat editorial changes and styling; S-100 WG4 decisions; references updated following publication of S-100 4.0.0, S-122, S-123.
1.0.0	Xxx 2019	S-100WG	First Edition.
1.0.0		J.Powell	Improved Readability

Page intentionally left blank

Contents

Part B - Steps in the Development of a Product Specification

1   Overview

S-100 the Universal Hydrographic Data Model is a 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 19000 series of geographic standards. This alignment enables easier integration of hydrographic data and applications into geospatial solutions. S-100 is inherently more flexible than S-57 and makes provision for such things as the use of imagery and gridded data types, enhanced metadata and multiple encoding formats. It also provides a more flexible and dynamic maintenance regime for features, attributes and portrayal via a dedicated on-line registry. S-100 provides a framework of components that enable the building of standardized product specifications for the modeling of hydrographic data, thus providing true interoperability between different data standards and systems.

S-97 is a Guideline intended for developers and maintainers of product specifications that utilize the IHO framework standard S-100 (Universal Hydrographic Data Model).

2   Introduction

The intent of Part B is to explain the various components of a Product Specification for those that are developing S-100 based product specifications. It describes the typical phases of development for the typical Product Specification, and a prescribed order for each phase. It provides guidance on how the rules contained in S-100 are used to build the components of a Product Specification, including documentation components, UML models, and machine-readable components such as XML Feature Catalogues and data formats.

3   References

ISO 8211	Specification for a data descriptive file for information interchange structure implementations. ISO/IEC 8211, 1994.
ISO 19109	Geographic information -- Rules for application schema. ISO 19109:2005. (Since replaced by ISO 19109:2015, but S-100 Edition 4.0 still references the 2005 edition.)
ISO 19115-1	Geographic information – Metadata – Part 1 – Fundamentals. ISO 19115-1, 2014, as amended by Amendment 1, 2018.
ISO 19115-2	Geographic information - Metadata - Part 2 – Extensions for imagery and gridded data. ISO 19115-2, 2009.
ISO 19115-3	Geographic information - Metadata - XML schema implementation for fundamental concepts. ISO/TS 19115-3, 2016.
ISO 19119	Geographic Information – Services. ISO 19119, 2016.
ISO 19136	Geographic information -- Geography Markup Language (GML). ISO 19136, 2007. (Also available as OGC 07-036 Geography Markup Language (GML) Encoding Standard. Open Geospatial Consortium Inc., 2007.)
ISO 19157	Geographic information – Data Quality. ISO 19157, 2013, as amended by Amendment 1, 2018.

ISO 19757-3	Information technology – Document Schema Definition Languages (DSDL) – Part 3: Rule-based validation – Schematron.
S-52	Specifications for Chart Content and Display Aspects of ECDIS / Specifications pour le Contenu Cartographique et les Modalites D'affichage des ECDIS
S-58	IHO S-58 ENC Validation Checks, Edition 6.0.0, May 2017.
S-99	IHO S-99 - Operational Procedures for the Organization and Management of the S-100 Geospatial Information Registry, Edition 1.1.0, November 2012.
S-99A	IHO S-99, Annex A - Conventions and Guidelines for the Content of the IHO GI Registry. (Under development)
S-100	IHO S-100 - Universal Hydrographic Data Model Edition 4.0.0, December 2018.
S-101	IHO S-101 - Electronic Navigational Chart Product Specification, Edition 1.0.0 (draft), July 2018.
S-111	IHO S-111 - Surface Current Product Specification, Edition 1.0.0 (draft), June 2018.
S-122	IHO S-122 - Marine Protected Areas, Edition 1.0.0, January 2019.
S-123	ISO S-123 – Marine Radio Services, Edition 1.0.0, January 2019.
SVGTiny	Scalable Vector Graphics (SVG) Tiny 1.2 Specification. W3C Recommendation 22 December 2008. URL: <http://www.w3.org/TR/2008/REC-SVGTiny12-20081222/>

NOTE: In this document, "S-100" means S-100 Edition 4.0.0 unless a different edition is explicitly identified.

4   Terms and abbreviations

4.1 Terms

abstract class

an object class which cannot be **instantiate**d, or is designated in an information model as not allowed to be instantiated

NOTE: Subclasses of an abstract class may be either abstract or non-abstract.

aggregation

special form of association that specifies a whole-part relationship between the aggregate (whole) and a component part (see **composition**) [ISO 19103]

application

manipulation and processing of data in support of user requirements [ISO 19101-1:2014]

application schema

conceptual schema for data required by one or more **applications** [ISO 19101‑1:2014]

association

semantic relationship between two or more classifiers that specifies connections among their instances [ISO 19103]

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) UML: feature within a classifier that describes a range of values that instances of the classifier may hold [ISO/IEC 19501:2005 (Adapted)]

(3) XML: name-value pair contained in an element [ISO 19136]

base64

an encoding designed to represent arbitrary sequences of octets in a form that allows the use of both upper- and lowercase letters but that need not be human readable [IETF RFC 4648 (restyled)]

code

representation of a label according to a specified scheme [ISO 19118:2011]

codelist

value domain including a code for a permissible value [ISO 19136]

codespace

rule or authority for a code, name, term or category [ISO 19136]

EXAMPLE: Examples of codespaces include dictionaries, authorities, codelists, etc.

composition

form of aggregation association with strong ownership and coincident lifetime as part of the whole [ISO 19103]

conceptual model

model that defines concepts of a universe of discourse [19101-1:2014]

conceptual schema

formal description of a **conceptual model** [ISO 19101-1:2014]

data client

an end-user receiving encrypted S-100-based data. The data client will be using a software application (for example ECDIS) to perform many of the operations detailed within the S-100 protection scheme [S-100 Part 15 (adapted)]

EXAMPLE: An ECDIS user.

data permit

file containing encrypted product keys required to decrypt the licensed products, normally created specifically for a particular **data client** [S-100 Part 15 (adapted)]

data server

an organization producing encrypted data files or issuing **data permits** to **data clients** [S-100 Part 15, (adapted)]

feature

abstraction of real world phenomena [ISO 19101:2003]

NOTE 1: A feature may occur as a type, class, or an instance. Feature type or feature instance should be used when only one is meant. **Feature class** should be used in the context of a model or Application Schema.

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 class

a class in an **application schema** or **model** that represents a **feature**.

feature type

an element in a **feature catalogue** that describes a **feature**, its **attribute**s, and **associations**.

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]

information type

an identifiable unit of information in a dataset with only thematic attribute properties [S-100 3-5.1.2 (adapted)]

instantiate

represent by a concrete instance [Merriam-Webster Online https://www.merriam-webster.com/dictionary/instantiate]

interface

named set of **operations** that characterize the behaviour of an entity [ISO 19119:2005]

metadata

information about a **resource** [ISO 19115-1]; data that defines and describes other data [ISO 11179-3:2013]

model

abstraction of some aspects of reality [ISO 19109-2015]

operation

> specification of a transformation or query that an object may be called to execute [ISO 19119:2005]
>
> NOTE: An operation has a name and a list of parameters.

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.

registry

> information system on which a **register** is maintained [ISO 19135]

relationship

> semantic connection among **model** elements [ISO 19103]

resource

> identifiable asset or means that fulfils a requirement [ISO 19115-1]
>
> EXAMPLES: Dataset, dataset series, service, document, initiative, software, person or organization.

scheme administrator

> organization solely responsible for maintaining and coordinating the protection scheme specified by S-100 [S-100 Part 15 (adapted)]

service

> distinct part of the functionality that is provided by an entity through **interfaces** [ISO 19119:2005]

spatial object

> object used for representing a spatial characteristic of a feature [ISO 19107:2003]

stream

> in online data exchange: a continuous sequence of fragmented data to be transported by a communication system [S-100]

universe of discourse

> view of the real or hypothetical world that includes everything of interest [19101-1:2014]

vocabulary

> terminological dictionary which contains designations and definitions from one or more specific subject fields [ISO 1087-1:2000]

4.2 Abbreviations

CORBA	Common Object Request Broker Architecture
CRS	Coordinate Reference System
CSS	Cascading StyleSheets
DCEG	Data Classification and Encoding Guide
ECDIS	Electronic Chart Display and Information System
ECS	Electronic Chart System

ENC   Electronic Navigational Chart

FC   Feature Catalogue

GFM   General Feature Model

GI   Geospatial Information

GML   Geography Markup Language

IALA   International Association of Lighthouse Authorities

IEC   International Electrotechnical Commission

IETF   Internet Engineering Task Force

IHO   International Hydrographic Organization

IMO   International Maritime Organization

ISO   International Organization for Standardization

JCOMM   WMO-IOC Joint Technical Commission for Oceanography and Marine Meteorology

MRN   Maritime Resource Name

OEM   Original Equipment Manufacturer

RENC   Regional ENC Coordinating Centre

SVG   Scalable Vector Graphics

W3C   World Wide Web Consortium

XML   eXtensible Markup Language

XSD   XML Schema Definition

XSL   eXtensible Stylesheet Language

XSLT   XSL Transforms

5   Overview of steps for development

When developing an S-100 based product specification it is recommended that developers consider each of the stages defined in the basic development process. The stages are outlined below and described in greater detail in the following sections.

5.1 Basic development process

The steps described below apply to the development of a Product Specification for a new data product.

1) **Initiation**. Identify the need for a new data product, define its scope, and decide the boundaries between the new product and existing data Product Specifications. Obtain sample source material. Describe typical application use cases.

2) **Develop the Domain Model/Application Schema**. Define the classes and attributes that describe the domain and which are relevant to the data product. Define the relationships between the classes and specify applicable constraints. Prepare one or more UML diagrams describing the Domain Model.

3) **Populate the GI Registry from the Domain Model**. Propose amendments to existing classes and attributes and propose new classes and attributes for addition to the GI Registry using the IHO GI Registry interface. Follow up on any returned proposals or queries from the Registry Manager or Domain Control Body.

4) **Develop the Feature Catalogue**. Prepare the XML Feature Catalogue from the feature and information classes, attributes and relationships as approved in the GI Registry, utilizing the Feature Catalogue Builder.

5) **Define data transfer modes and packaging**. Determine whether data products are to be delivered as data files contained in transfer (exchange) sets, by web services (and if so, identify or outline a service protocol), e-mail, etc. Determine whether data is to be delivered in real or near real-time. Identify constraints and requirements arising from delivery mechanisms and communication constraints such as message size, bandwidth limitations, availability of communications to customers, licensing and payments, encryption, etc.

6) **Define metadata**. Survey the metadata elements listed in S-100 for their appropriateness to the data product and its allowed packaging and delivery methods. Define appropriate values and restrictions for the metadata elements listed in S-100. Consider whether additional product-specific metadata elements are needed.

7) **Define the data format**. Select an appropriate data format. S-100 provides for 3 standard delivery formats (ISO 8211, GML, and HDF5). Define format-imposed items (for example embedded header metadata). Prepare format-specific artefacts if necessary (for example GML "application schema" XSD files for the GML format).

8) **Develop the Data Classification and Encoding Guide (DCEG)**. The DCEG is intended for cartographers, editors, and data encoders, rather than application developers or OEMs, and should be written from that perspective. A DCEG should contain enough overview and general material about basic concepts such as data types, features, information types, associations, etc, to give its intended audience a basic grounding in the concepts they will need to apply; but its main focus should be on what to encode in the data product, and how to encode it.

9) **Define portrayal symbols and rules**. Determine the symbols to be used for portrayal and the rules for generating displays from the data product. This means deciding which features should be displayed; which combinations of attribute values or associated data objects should be assigned distinct symbols; and which pre-existing symbols and colours can be reused and which need to be defined. It also includes defining the new elements in the appropriate formats.

10) **Registration of portrayal elements**. Propose any new portrayal components (for example symbols, colour tokens, line styles, area fills, etc) to the portrayal register in the GI Registry using the IHO GI Registry interface. Follow up on any returned proposals or queries from the Registry Manager or Domain Control Body.

11) **Develop the Portrayal Catalogue(s).** Prepare a Portrayal Catalogue (or Catalogues) for the features and information types which are intended to be displayed in the intended application domain(s) and usage scenario(s). This consists of encoding the rules in the appropriate portrayal language using the symbols determined in step (8).

12) **Define the spatial reference system**. Identify the recommended coordinate reference system and vertical datum(s).

13) **Define data product packaging and maintenance**. Define the content and structure of delivery packages, updating of data, and any auxiliary content delivered either with or as an adjunct to data.

14) **Define validation checks and quality measures**. Define tests for the spatial, structural, and conceptual integrity of datasets. Define format-specific implementations of validation checks (for example Schematron rules [ISO 19757-3] for the GML format). Define quality measures for the datasets (quality measures will generally be statistics about passed and failed tests).

15) **Determine interoperability** with other data products. Determine which if any product groups in Interoperability Catalogues are supplemented or enhanced by the data product; and how the IHO Interoperability Catalogue will be affected by the new product, including updates to display priorities, interleaving, predefined combinations and other interoperability rules and operations.

16) **Prepare sample data** for test-beds. Create sample datasets and exchange sets conforming to the data format, packaging, and Feature Catalogue defined in the Product Specification.

17) **Testing and feedback**. Carry out tests of data production and use of the sample data in selected applications to validate the correctness, completeness, consistency, and utility of the Product Specification, including related artefacts such as the Feature Catalogue and XML schemas.

Development of Product Specifications is an iterative process of continuous refinement as it moves through the S-100 Readiness Levels. Developers should expect to cycle through the stages above multiple times, in smaller or larger cycles depending on experiences and results of each stage. For example, the first preparation of sample data (Stage 16) might identify gaps in the Domain Model and require revisiting Stage 2 (development of the Domain Model); or the development of portrayal rules (Stages 9 - 11) may identify gaps in the Domain Model (Stage 2). Passage through some intermediate stages in the cycle may be trivial on the second and later passes, as the results may be unaffected by discoveries during later stages in the process – for example, packaging and delivery or choice of spatial reference systems are likely to be unaffected by changes identified during the preparation of sample data.

Figure 5-1 depicts the development process, including probable cycles.
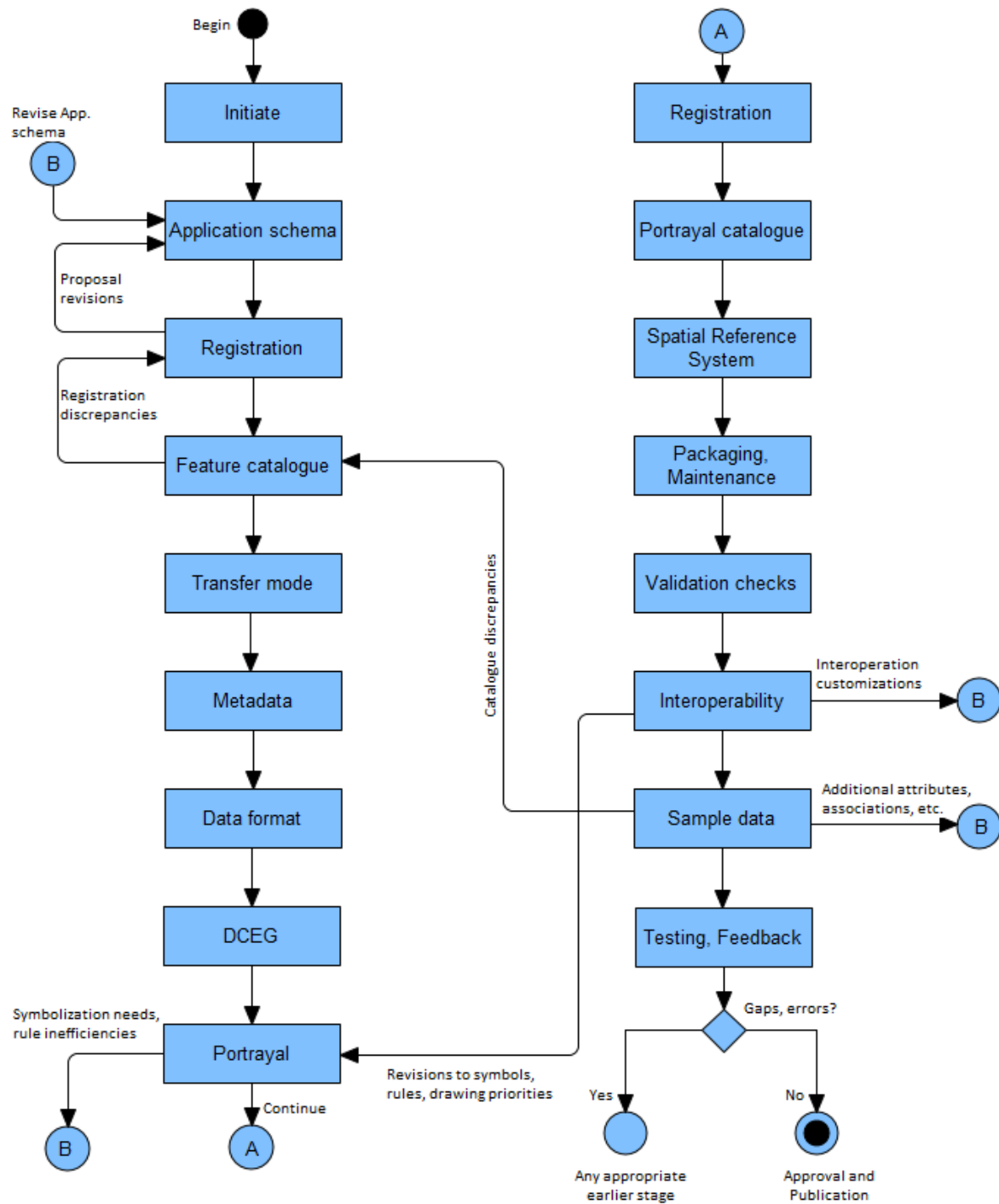
Figure 5-1 - Product Specification development process

S-100 Part 11, Appendix 11-A describes an idealized process (Figure 11-A-1 – Product specification process) for the basic mechanical steps in developing a domain model plus related registry actions. The figure also includes black-box stages for "Coordinate Reference System" and "Product Specification Documentation" at the end of the process. The actual process is an iterative refinement process.

5.2 HSSC Review Cycle for S-100 Based Product Specifications

IHO Resolution 2/2007 outlines the process for development and approval of IHO based product specifications. As part of this lifecycle it is important that working groups and project teams engage with the IHO's Data Quality Working Group to review product specifications for adherence to S-100 data quality aspects and completeness. While this review can occur prior to Edition 1.0.0 (Testing and Implementation Edition), DQWG will review and provide feedback during the Edition 1.0.0 to 1.9.n phase specification development. Edition 1.0.0 of any IHO S-100 based product specification is not designed for regular use in approved arrangements or for provision of operation services by purpose. Edition 2.0.0 of a standard is considered to be mature and can be used for implementation. The following figure shows the IHO Review Cycle as approved in IHO resolution 2/2007.

Proposed Review Cycle for WG/PT Development Phase
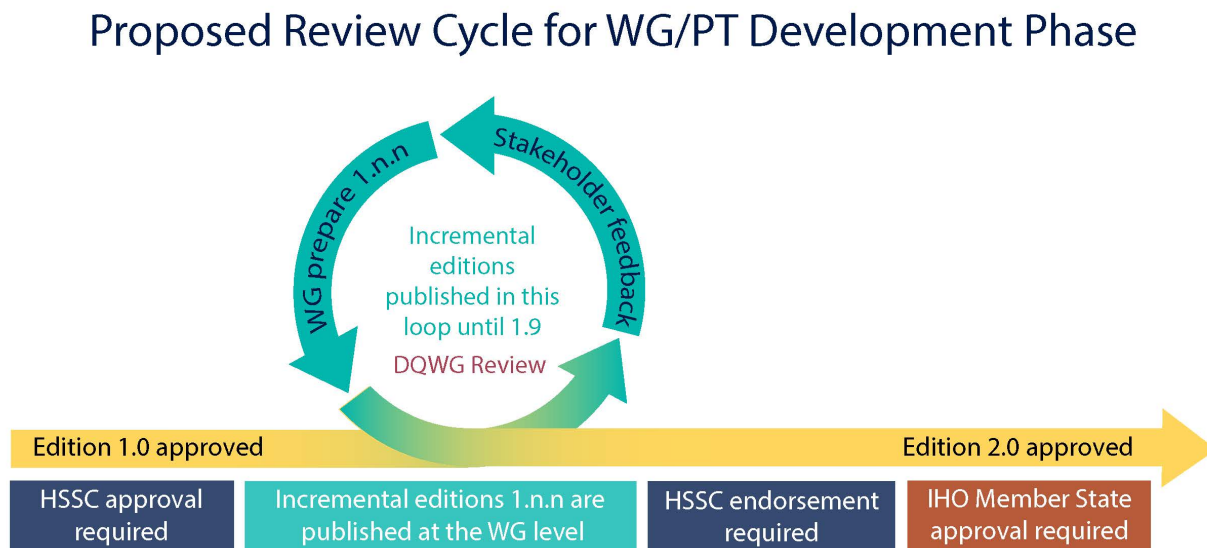
Figure 5-2 - Review Cycle (IHO Resolution 2/2007)

5.3 Process for extending a specification

If there is a demonstrated need to extend an existing product specification, developers should review the steps in the basic process in section 5.1 and determine which can be omitted.

For example, if features, information types, attributes, or relationships are added, removed, or modified, most of the steps in the basic process will be needed. The spatial reference system and packaging/maintenance may not change in such a revision and the corresponding phases can be skipped.

If the extension consists of defining a new transfer mode, such as adding a streaming delivery mode for data products originally intended for exchange set delivery, it is likely that only the following stages need be executed:

- Transfer mode – to add the streaming mode for transfer, define specifications for servers, etc.
- Metadata – to define which metadata elements in the exchange set mode apply to the streaming mode, whether any new metadata elements are needed, and how metadata is delivered or made available in the new mode.

- Data format – to consider whether the already defined data format is appropriate for streamed transfer and compatible with the intended service delivery and protocols, and specify an appropriate format if necessary.

- Packaging and maintenance – to specify the appropriate data wrapper formats and service protocols; for example WFS, REST, etc. Defining new wrapper formats or protocols is permitted but should be the last resort.

- Sample data – preparation of sample data for testing the streaming mode.

- Testing and feedback – this should include a prototype or simulated streamed data transfer environment.

6   Initiation

The initiation phase consists of the following activities. Except for the first (identifying the need) most of the other activities can be done in parallel.

1) Identify a need that a data product will fill – this will normally be the result of external circumstances such as definition of a service portfolio, application-driven demand for new kinds of information, new legal requirements for shipping, IMO decisions, etc.

2) Define the scope of the product – the subject area, the kind of information it is expected to contain; and equally important, what information it will not contain.

3) Determine if existing Product Specifications can be extended. If so, such an extension will probably consume less time and effort than developing a new Product Specification.

4) Determine sub-areas within the product; that is, the scopes within the Product Specification, or a new scope for an existing Product Specification.

5) Define constraints – domain, application, platform, etc.

6) Collect samples of source information. These generally include existing databases, official, unofficial, government, and commercial publications – especially those in wide use.

7) Define application use cases.

8) Outline application functionality enabled by the data product.

9) Define delivery modes (transfer set, messages, web services, etc).

10) Obtain approval from the appropriate sponsoring organization.

11) Put together a Project Team.

If the product specification development is performed under the umbrella of the IHO then the following step also needs to be included in the initiation stage:

Before deciding to include a new S-100 based product specification in the HSSC work program, the following considerations should be taken into account:

1.   Has the need for the product specification proposed been documented?
2.   Has a compelling need been demonstrated with detailed examples?
3.   Has the product specification been considered within the scope of HSSC?
4.   Has the description of the proposed product specification sufficiently addressed the cost to the IHO and possible legislative and administrative burdens?
5.   Has the development duration been considered?

To simplify the above mentioned decision making process, the Product Specification description should provide a set of information beforehand to HSSC for consideration.

Product Specification description		
Action	**Description**	**Comments**
Product Specification number	The intended number for this product specification	Keep this section empty until HSSC have assigned a number
Title	The title of the product specification	
Abstract	A brief summary of the product specification summarizing: 1. the intended use 2. the primary and secondary user 3. the expected functionality	
Product Specification Scope	The overall scope of the specification	This should include the expected SOURCE(s) of the data. For example: NIPWG publications...a source would be the Coast Pilot/Sailing Directions for S-126. (that way if someone thinks tide tables should be here instead of with S-104 it will be listed)
Justification	The reason why this product specification should be developed	
Specification Interoperability	Any interoperability with other product specifications within the S-100 family	Specify the limits of the product specification
S-98 Applicability	Applicable to S-98 (Yes or No)	S-98 Interoperability Specification in S-100 Navigation Systems
Cooperation with other HSSC WGs	Specify which WG will be involved to which extend	NCWG: Portrayal DQWG. Specific information on Data Quality
Budget	Statement of budget need and the figure	Is IHO budget needed? Will the development be financed by an

| | | external party and to what extent?

There should be general guidelines for how to calculate this...with reasonable values based on reality and evidence. |
|---|---|---|
| Schedule | Description of the intended time frame | Or at least the steps to accomplish since this is so very difficult to really determine. |

Table 6-1. Product Specification description template

7   Develop data model (Application Schema)

7.1 Introduction

The Application Schema as defined in S-100 is usually synonymous with "domain model". It is a specification of the classes, attributes, and relationships relevant to the data product.

A Product Specification will include only a single Application Schema (which may be broken up into multiple diagrams). However, in theory a Product Specification that describes different scopes may need to distinguish Application Schema for different scopes – for example a product that includes both vector and coverage data might need two Application Schemas.

To minimize complexity, Product Specification developers should try to avoid defining multiple Application Schemas.

Note that this does not prevent a single Application Schema from being depicted using multiple diagrams.

The Application Schema should describe only features, information types, their attributes and relationships that are part of the data product. Any other classes, constraints, or elements that are used to further understand the data product and its role in applications, services, or service portfolios should be distinguished from the Application Schema and documented separately in the Product Specification.

Model developers should try to limit the number of model elements while still allowing implementations to make appropriate (conceptually appropriate, logically appropriate, consistent, correct, and performance-based) distinctions.

The principles of data normalization learned in relational database design should be kept in mind, but model developers should also note that an S-100-based domain model and applications schema are feature-attribute-relationship models and not database designs.

7.2 Steps in model development

1) Determine whether the data product is coverage or vector data. Coverage data is characterized by values of characteristics distributed over an area or areas, while vector data is characterized by localized regions (points, and/or areas) that possess boundaries and do not exhibit internal variation in characteristic values (or where such internal variations can be ignored).

2) Identify the concepts in the application domain. This will involve reviewing the source material to identify important features and information in the domain which will be useful to end-users. Source material will include the sample texts identified in the initiation phase and, if available, documentation and data dictionaries of relevant applications, requirements for existing applications, related standards and circulars from IEC, IMO, IALA, etc.

3) Search for existing concepts using key words (classes, attributes and relationships) in the IHO GI Registry which can be re-used.

4) Develop new concepts only for those that do not yet exist in the Registry. This will involve examining the source material mentioned earlier in more detail to pin down concepts and their definitions. The process for submitting proposals is described in S-99.

5) Define the classes and attributes that describe the domain and are relevant to the data product. If classes, attributes and codelists/enumerations are already defined in the Registry or existing Product Specifications they can be reused. If not, the Project Team will need to develop and define classes and attributes, including listed values for enumeration and codelist types.

6) Define the relationships between the classes. Relationships should be defined in order to capture those that exist in the real world and to make links which are useful for application processing. Both reasons will often apply.

    EXAMPLE 1: A structure/equipment relationship between classes modelling structure objects and classes modelling equipment mounted on the structure.

    EXAMPLE 2: A contact information relationship between classes modelling pilot service areas and contact information for pilot services available in that area.

7) Specify any constraints applicable to the classes, attributes, and relationships. Examples are constraints requiring conditional encoding of attributes, exclusive relationships (that is, when an instance is allowed to participate in only one of multiple possible relationships), etc. Generally, structural restrictions are depicted in UML class diagrams, while value restrictions on individual attributes are not (to reduce clutter). Whether depicted or not, any restrictions should be enforced and documented in the appropriate section or artefact of the Product Specification.

8) Prepare one or more UML class diagrams describing the domain model. Recommended practices for S-100 models are based on ISO TC211 recommended practices as modified by Section 7.6 of this document.

9) Prepare supporting text explaining the overall structure of the Application Schema and for each diagram explaining the purpose of each diagram and the relationships between the classes. In addition, explain any additional subtleties of the classes or their relationships that may not be obvious or should be specially noted. This text should not be a listing of classes, attributes, and relationships, but instead should clarify the purpose of the model depicted in the diagram by explaining what domain phenomenon the diagram captures and how the classes and relationships express it. For example, from S-122:

    Some protected areas require reports to be filed with authorities when certain events occur such as an animal strike or pollution event. Other areas require reporting to specified authorities when entering, leaving, etc. These requirements are modelled by association of a ShipReport class to the Authority class. The area in question is modelled by a feature of the requisite type; for example., a MarineProtectedArea or VesselTrafficServiceArea, as described in clause 6.2.1.3 [of S-122]. Any time

requirements or constraints on the filing of the report are described by the noticeTime attribute, with explanations, if any provided in text form in the textContent attribute of ShipReport. Required reporting formats, if necessary to be included, are also described in the textContent attribute. [Figure 7-1] shows the model elements that are used to carry these conditions, note that not all permitted associations or roles are included, in order to reduce clutter.
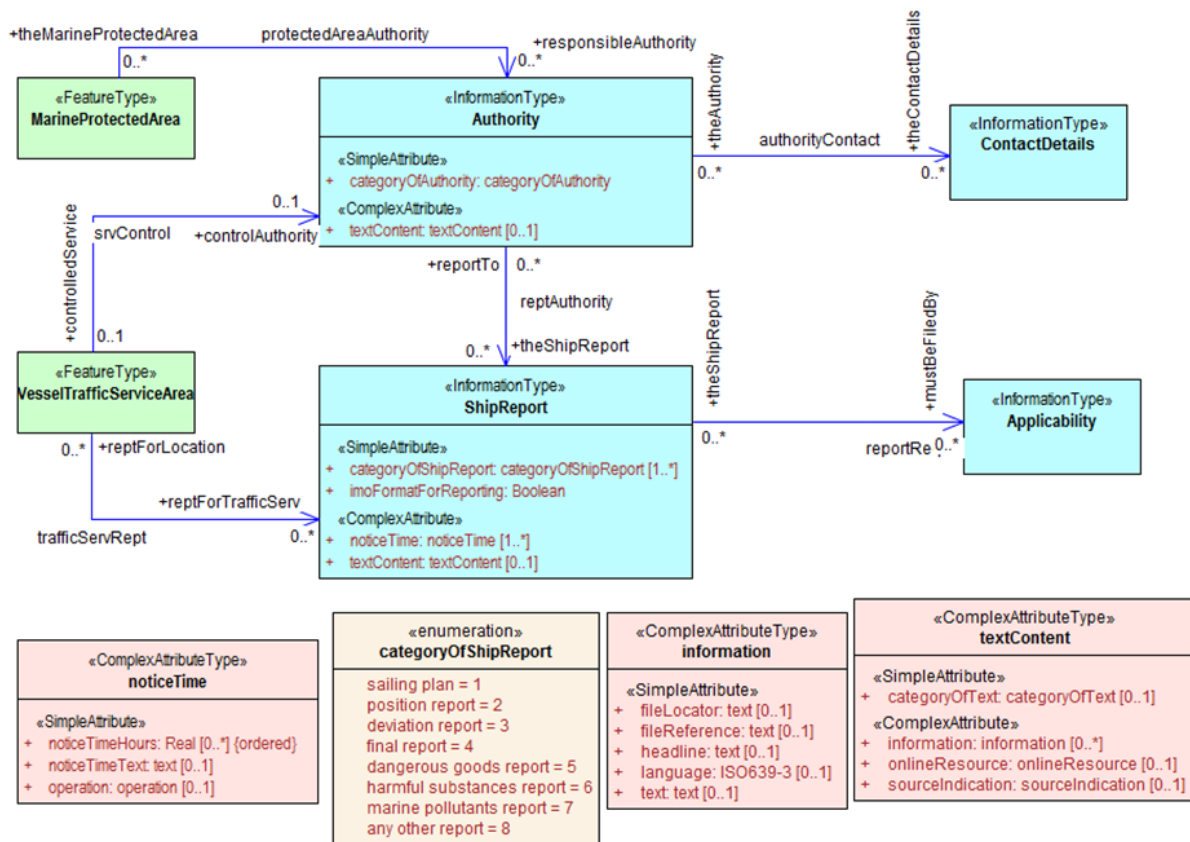
Figure 7-1 - UML diagram depicting part of the S-122 Application Schema (from S-122)

7.3 Relationship to the General Feature Model

The General Feature Model is a conceptual model for feature, their characteristics and associations and acts as the basis for the structure of feature catalogues. Feature and information types in the Application Schema must be realizations of the meta-classes S100_GF_FeatureType and S100_GF_InformationType from the S-100 General Feature Model (GFM) (S-100 Part 3, Figure 3-1); or subclasses of a class that realizes the appropriate meta-class. Attributes must be realizations of the thematic or spatial attribute meta-classes defined in S-100 Part 3, Figure 3-2; or subclasses of a realization. Product specifications may define local root classes from which all their feature and information classes are derived, as shown in the Figure below, or may realize feature and information type classes from S100_GF_FeatureType and S100_InformationType.
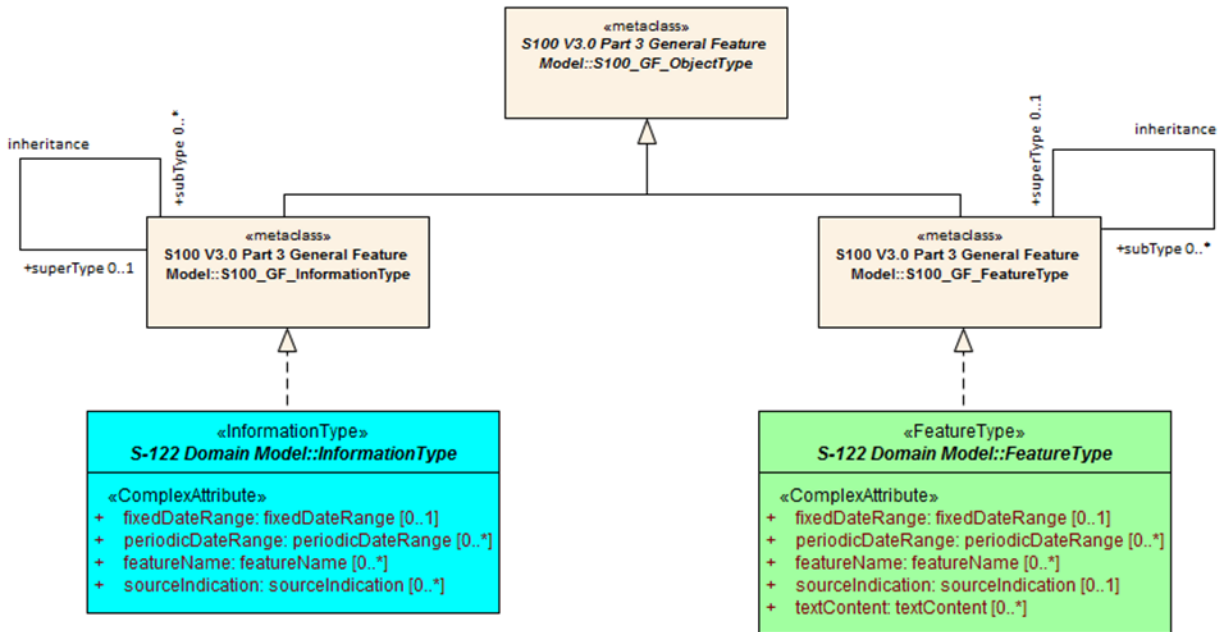
Figure 7-2 - Example of realization from S-100 GFM

7.4 Rules for Application Schemas

S-100 rules for Application Schemas are based on ISO 19109. The S-100 rules for Application Schemas are defined in S-100 Part 3, clauses 3-6 and 3-7.

7.4.1   Application Schemas for vector data

Application Schemas for vector data should follow the guidelines as described below. S-100 Part 3, clause 3-6 describes the rules for applications schemas in greater technical detail:

- Features, information types, and complex attributes must be modelled as classes.

- Relationships are modelled by UML associations or association classes (the latter only when the association itself is characterized by attributes – see clause 7.5.4.2 in this document).

- Attributes are modelled by UML attributes in the appropriate class.

- Associations must be labelled (have association names). Navigable association ends must also be labelled (should have role names).

   NOTE: Diagrams may suppress depiction of labels for clarity and to reduce clutter. Labels may be defined by specific rules given in the Product Specification text instead of the UML diagram (for example a Product Specification is allowed to 'label' an association end using a statement like "The role of *FeatureX* in all its associations is *theFeatureX*" (see S-100 Part 3, clause 3-5.4.5 on default names for association ends).

- Spatial attributes must be modelled either as attributes with data type one or more (that is, union) of the allowed spatial types in the spatial schema, or an association between the class that represents a feature and one of the spatial objects defined in the spatial schema.

- Enumeration types and their listed values must be modelled by UML enumerations; codelists must be modelled as UML classes with tags specified in S-100 Part 3, clause 3-6.7.

- Standard schemas (for example the spatial schema, Feature Catalogue schema) must not be extended within Application Schemas.
- All classes used within an Application Schema for data transfer shall be instantiable. This implies that the integrated class must not be stereotyped <<interface>>.
- A UML Application Schema must be described within a UML package, which shall carry the name of the Application Schema and the version stated in the documentation of the package.

7.4.2 Application schemas for coverage data

The rules for Application Schemas for coverage data are described in S-100 Part 3, clause 3-7; and Part 8.

The rules are similar to the rules for vector data, with the following difference:

- Spatial types for coverage features are modelled by the appropriate point set, grid, or TIN type defined in S-100 Part 8.

7.4.3 Additional rules

Names of features and information types must use their camel case codes.

Vector feature classes must use the stereotype <<FeatureType>> and information classes must use the stereotype <<InformationType>>. A stereotype allows designers to extend the UML model by creating new model elements.

Coverage *type* elements (describing the coverage geometry) compliant to S-100 should use the appropriate stereotype from S-100 Part 8, and Application Schemas for coverage data may depict the data attributes by defining a <<FeatureType>> element with the thematic data attributes and associating it with the coverage type element.

Figure 7-3 - Example of coverage feature and type elements conforming to S-100 (from S-111)

If necessary, Product Specifications may use domain-specific stereotypes in addition to the standard stereotypes.

Abstract classes are indicated by italicizing the class name (Enterprise Architect does this automatically if the "Abstract" checkbox is checked in the UI).

S-100 states that "the use of multiple inheritance shall be minimized, because it tends to increase model complexity". Multiple inheritance is the situation where a class has more than one immediate superclass. Application Schema developers should note that multiple inheritance contravenes the S-100 GFM, which allows feature and information types to have at most one super-type.

7.5 Other conventions and recommendations

7.5.1   Reuse and harmonization

Before new elements are defined the IHO GI Registry should be checked for existing elements that can be re-used. The following should be also considered:

- Features, information types, and attributes should be re-used whenever possible.

- Structure and associations should be harmonized with S-101 and other existing related or complementary products.

- Defining similar but slightly different items should be avoided unless absolutely essential.

- Extensions such as additional listed values in an enumeration can be proposed to the Registry, but conflicts such as different definitions for the same terms must be avoided if at all possible.

- Existing items may be re-used with the addition of product specific constraints, such as limiting the set of allowed values for an enumeration or codelist type. Such harmonization includes, for example, re-using complex attributes defined in other Product Specifications with restrictions that exclude some of their sub-attributes.

7.5.2   Features and information types

A feature class is "an abstract representation of real-world phenomena" (S-100 Part 3, clause 3-5.1.1). The Application Schema should define a feature class for describing:

1) A concept whose instances have a spatial (geographic) location. Such a class represents a **geographic** feature.

2) A concept whose instances are collections of the above. The collection may consist of instances of one such class or several such classes. Depending on the nature of the collection, the concept will be either an **aggregation** feature or a **composition** feature. In Application Schemas they are treated like geographic features but may or may not have a spatial attribute.

3) Cartographic information (such as feature names, labels, compass roses, legends) that are intended to be positioned at a specified location (or must re-positioned from its default position relative to a feature) to avoid obscuring other features. Though not a representation of any real-world phenomenon, such information is also modelled as a feature and treated as a feature in the Feature Catalogue. Such features are called **cartographic** features. The information to be displayed may be encoded in an attribute of the cartographic feature class or identified by reference to another feature class.

4) Meta-information or metadata pertaining to all features (or defined subsets or defined attributes of features) in a particular area. These are called **meta** features.

The S-100 Feature Catalogue model provides a *featureUseType* element for designating feature types as "geographic", "cartographic", or "meta". Aggregation and composition feature types are designated as being of "geographic" *featureUseType*.

Information types are identifiable objects that can be associated with features or other information types, in order to carry information particular to the associated objects. Information types may be considered classes that are typically used to share information among features and other information types. Information types have only thematic attribute properties.

7.5.3   Superclasses and subclasses

Defining abstract superclasses is recommended when 3 or more conceptually similar classes exist in the model; the similar classes have some of the same attributes or relationships; and the allowed values of shared attributes are the same. It is not necessary that the classes bind exactly the same sets of attributes or have exactly the same relationships.

Subclasses inherit both associations and attributes, unless explicitly overridden. While overriding is permitted, it introduces additional complexity in data formats and implementations. Instead of overriding, the model should be changed to move the attribute or association down in the hierarchy and assign it only to classes where its use is permitted.

7.5.3.1   Subclasses versus category attribute

There are different ways of modelling the same item. For example, in order to indicate different buoy types, either the subclass approach or the enumeration approach (categoryOf) can be used. Both approaches are depicted in Figure 7-4.
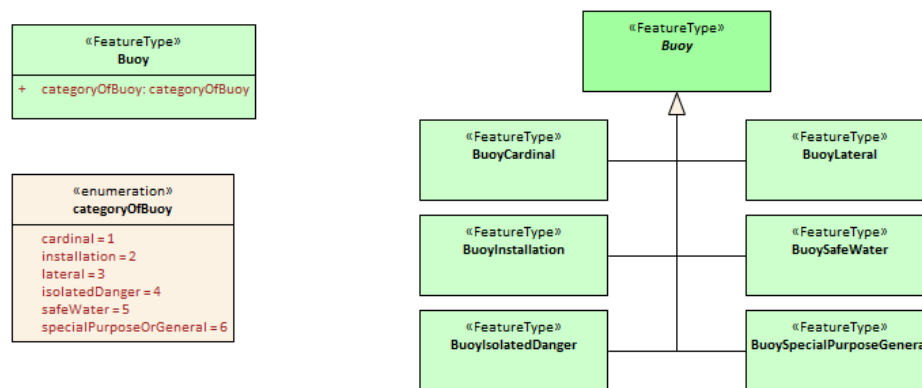
Figure 7-4 - Illustration of alternative models using category attribute and sibling subclasses

It is up to the product specification development team to determine which approach best suits their needs. In order to determine this consideration should be given to the following issues:

1) Will either approach result in a significant divergence from some external source?

2) Will the "category attribute" approach cause issues for portrayal because the symbols for the different types are presumably different? Or will the subclass approach result in unnecessary portrayal rules because the symbols are the same?

3) Will any of the subclasses have its own specific attributes or relationships? If so, the sub-class approach is preferable.

4) Are the different categories/subclasses in (or likely to be placed in) different viewing groups, or have different drawing order? If so, there is a slight preference for making subclasses. (Only "slight" because the portrayal rules and Interoperability Catalogue can use attribute values in assigning viewing groups to feature instances.

5) If subclasses are used, will that introduce situations where it may be necessary to encode coincident objects with different categories? The answer "Yes" suggests a preference for the categoryOf... approach.

6) Are the subclasses conceptually very different? "Yes" implies the sub-classes approach.

7) Which approach is likely to be compatible with external resources like existing databases and implementations?

8) If there are a large number of subtypes, then the categoryOf... approach may be preferable because it leads to more compact representations in UML diagrams and more compact DCEGs. ("Large" is obviously subjective, but will generally between 5 and 9 based on research into human cognitive psychology and probable implementation methods in user interfaces – there will be variations dependent on concept semantics and similarities).

9) Overall complexity of the Application Schema and Feature Catalogue. Sibling subclasses of features (or information types) generate more artefacts and documentation than a category attribute. They certainly mean an additional table for each subclass in the DCEG; an additional XML element for each in the Feature Catalogue; and a box in the UML diagram Application Schema for each class. To that extent sibling subclasses are a greater cognitive burden on encoders and developers.

7.5.4   Associations and association classes

7.5.4.1   Navigability, source, and target

Association navigability should be indicated if the association is navigable in only one direction; that is, the model designer expects applications to access one object from the other, but not vice versa. Feature/feature and information/information associations are usually navigable in both directions, while feature/information associations must be navigable from the feature end but are not required to be modelled as navigable in the other direction. UML regards navigability information in UML diagrams as hints to implementations rather than hard requirements, and implementations and data formats are free to implement navigability in the most efficient manner.

Unidirectional navigability will normally also determine the source and target of the association.

An association's source and target should be grammatically and semantically compatible with the name and definition of the association; for example, for the association Person/subscribes/Magazine the source should be Person and the target Magazine.

For feature/feature associations both ends should be named; for feature/information associations the information end should be named and the feature end may be named. This is an S-100 requirement, not a UML requirement.

7.5.4.2   Association classes

Association classes are a means of adding parameters (characteristics) to associations, rather than to the classes at the end of an association. An attribute of the association class characterizes the <u>relationship</u> between the classes at the ends of the association.

The use case for association classes is basically "whenever a relationship is characterized by one or more attributes."

EXAMPLE 1: A specified set of vessels is COVERED by a regulation and another set of vessels is EXEMPT from the regulation. The sets of vessels are described by an information type class; the regulation by another information type class; and the relationship between them by an association class which has an attribute characterizing the relationship as inclusion or exclusion (of the specified subset in the specific regulation). This relationship is modelled by the **InclusionType** association class in Figure 7 5.

EXAMPLE 2: Vessels with specified cargo and dimensions must use a specified pilot boarding place; vessels of smaller dimensions are recommended to use the boarding place; and warships are exempt from using

the pilot boarding place. The sets of vessels are described by an information type class; the pilot boarding place is a feature class; and the relationship between them by an association class which has an attribute stating whether the specified set of vessels is required/recommended/exempt from use of the pilot boarding place. This relationship is modelled by the **PermissionType** association class in Figure 7 5.
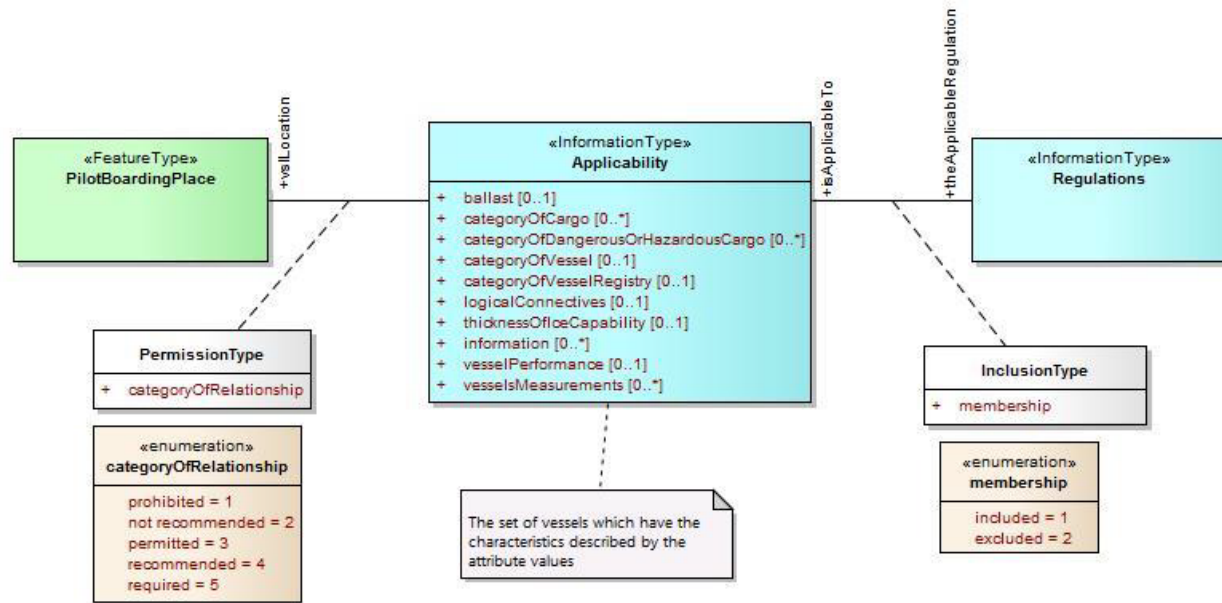
Figure 7-5 - Examples of modelling with association classes

7.5.5   Attributes in general

Certain attributes may use the same set of listed values as other attributes. For example, an enumeration for compass points may be shared by the attribute **windDirectionCompassPoint** which is the direction for where wind is coming from, and **directionOfMovement** which is the attribute describing where a weather system is going toward. At present this can be simply handled in the attribute's definition.

Complex and spatial attributes can be modelled as either named attributes in the UML model class element with a type corresponding to the spatial primitive or the name of the complex attribute; or alternatively, separate model elements linked to feature/information class by an association (ordinary association for spatial type, composition for complex attributes). The two methods are illustrated in Figure 7-7 below. The second method is not suitable for complex models due to the additional boxes and association lines.
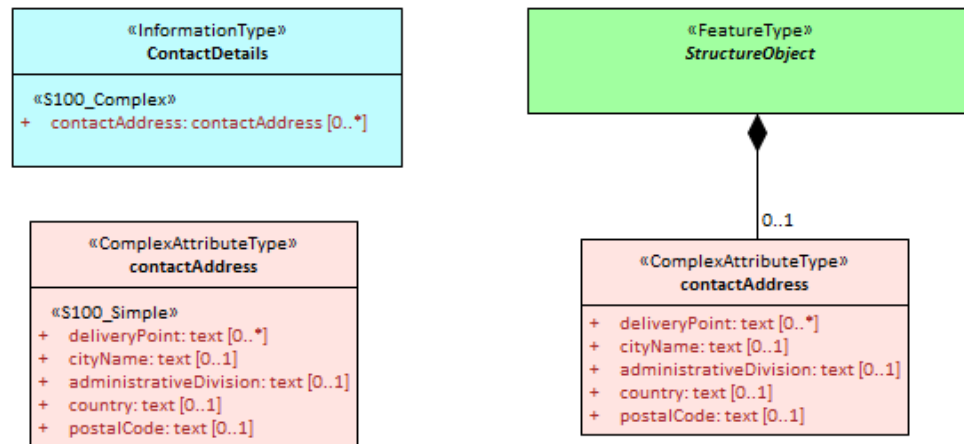
Figure 7-6 - Methods of representing complex attribute bindings

7.5.6   Codelist and enumeration attributes

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 Product Specification authors or the list of allowed values is long, volatile, controlled by another authority, and/or shared by multiple domains).

Individual members of a codelist or enumeration ("listed values") are generally meaningful only in a given context (or "container"), which is a specific (named) enumeration. The context corresponds to an attribute concept.

7.5.7   Labels and definitions for listed values

Labels should be short but informative, keeping in mind that both end-users and encoders may leverage them. Implementers of end-user display and production tools are likely to use listed value labels as 'tooltips' or explanatory text, or as the 'display text' of the attribute numeric code value for end users. End-users are more likely to see labels than full definitions due to other demands on their attention and screen display constraints.

7.5.8   Data types

S-100 defines a set of primitive and derived data types (Part 2a, clause 2a-4-10). Attribute values of thematic attributes should be one of the types listed in that table. If restrictions on the values are needed, Product Specifications may define constraints, if possible encoded using one or more of the elements in S100_CD_Constraints (length, range, pattern, and precision). Constraints which cannot be encoded must be documented as a 'Remark' or note.

The data types defined in S-100 can in principle be extended by Application Schemas but if this is done the Product Specification must define the extended data type in terms of the predefined data types in S-100 and use the predefined data type in the Feature Catalogue. Data formats may use their equivalent built-in types which are defined in the underlying format standard (for example HDF5 and XML built-in types) in order to leverage standard data validation software provided the equivalence is documented either in the Product Specification or the underlying format standard.

7.5.9 Codes for listed values

 Numeric codes *must* be positive integers, and *should* be in the range 1-254 if possible (to allow data formats and implementations to use compact representations – but codes up to 65535 are allowed). Codes used for retired listed values can only be used if the proposal is a revision (supersession) of the retired listed value.

The Registry also allows specification of alias and camel case codes for listed values. The camel case field should be completed with a camel case code derived from the label, beginning with a lowercase letter. The guidelines for camel case codes are specified in S-99 Annex A.

7.6 Recommended practices

7.6.1 Reviews of model elements and structure

Models should be reviewed at regular intervals while under development, with reviews involving domain experts as well as information modelling experts.

7.6.2 Diagram layout

Common 'best practices' for layout of UML diagrams should be followed. In particular, diagrams should not contain too many elements; should minimize line crossings; and use vertical layouts for hierarchies (or left-right horizontal layouts if a vertical layout does not work). Lines representing associations should minimize the use of curved segments.

7.6.3 Colour coding of model elements

Colour coding should be used to distinguish diagram elements for features, information types, enumerations and codelists, complex attributes, association classes, and constraints and notes. Abstract types should be indicated by darker shades.

Figure 7-1 and Figure 7-6 illustrate the use of colour coding to depict different kinds of UML elements. Compare the shades of the non-abstract feature and information classes in these figures to the abstract feature and information classes in Figure 7-2.

S-100 departs from ISO TC211 recommendations for the use of black-and-white-only UML diagrams in order to distinguish between feature and information types (the concept of information type is unique to S-100).

7.6.4 Documentation tables

This may be formatted like the UML schema documentation tables in S-100 or generated by the UML software. Whichever method is used, the documentation must document the classes, attributes, enumeration and codelist types; and associations in the Application Schema, including names, definitions, multiplicities, data types and roles.

7.6.5 Recommended Software Tools

The S100 Working group recommend using Enterprise Architect™ to develop the UML Application Schema(s). Other UML tools or special templates in off-the-shelf editors may also be used but are likely to have minor differences in UML notations which will need to be adjusted or explained in the Product Specification.

XML data including Feature Catalogues and metadata is easier to view in open-source or COTS XML software rather than ordinary text editors.

7.6.6   Identification of models

The identification of each Application Schema shall include a name and a version. If there is only one Application Schema in the Product Specification, this identification is implicit in the name and version of the Product Specification. Product specifications with more than one Application Schema must identify each, potentially by associating it with a scope.

8   Data Classification and Encoding Guide

The Data Classification and Encoding Guide (DCEG) provides information on how the data is to be captured. This should be as detailed and specific as necessary. The DCEG is used primarily by cartographers, editors, and data encoders, rather than application developers or OEMs, and should be written from that perspective.

The DCEG includes the collection criteria for mapping real world objects to the conceptual objects of the dataset.

Any organization performing data capture for the data product defined by the 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.

9   IHO GI Registry

Procedures for registration are explained in S-99. It is recommended that at least one member of the Project Team or Working Group register as a Submitting Organization. Submitting organizations propose changes and additions to the contents of Registers. Submitting Organizations will normally represent a recognized body or stakeholder group (such as from government, industry, academia, and relevant user groups). Registered submitting organizations may submit proposals for consideration under any Domain in a Register. Stakeholders and any other interested parties who do not wish to enrol should submit proposals through an existing Submitting Organization.

To harmonize with other Product Specifications, propose extensions to Registry items where possible; for example propose generalization or specialization of an existing element, or additional values in an enumeration or codelist type. Restrictions of existing types can become new sub-types rather than changes to an already defined type; or it may suffice to define a constraint in the Product Specification.

10 Feature Catalogue

The Feature Catalogue is an XML document which conforms to the S-100 XML Feature Catalogue Schema. Note, for Imagery and Gridded Data, a coverage is a feature type, and a Product Specification Feature Catalogue should define the attributes, coverage feature (with spatial primitive type 'coverage').

Feature Catalogues should be documented by a text-based documentation of their contents, which should also be reviewed by the Project Team and responsible Working Group. For review purposes, this text-based documentation should be generated from the XML Feature Catalogue. The resultant text can be in Word or PDF, or another format preferred by the team.

11 Data transfer modes and packaging

Define mode of delivery – exchange set, message, or service. A Product Specification can also specify more than one delivery mode.

This needs to be done before metadata is specified because some metadata elements as well as the treatment of metadata (for example separate vs. embedded) depend on delivery mechanisms, constraints, and protocols.

Details of packaging and transfer content can be finalized in a later stage (see section 17).

12 Metadata

The minimum metadata requirements are set forth in Part 4 of S-100 (Appendix 4a-D for vector data, Parts 4b and 8 for coverage data). Product Specification developers should consider whether the metadata elements listed in S-100 are relevant to the data product and which of them are appropriate for its allowed packaging and delivery methods. For relevant elements, define appropriate values and restrictions if necessary for the metadata elements listed in S-100 Appendix 4a-D or Parts 4b and 8. Developers should note that Part 4b is quite skeletal in S-100 Edition 4 and the development team will need to use the underlying ISO standards and ISO metadata schemas.

12.1   12.1 Metadata for exchange set products

This section describes metadata for products that are delivered in exchange sets.

12.1.1 Generic metadata model

S-100 provides for S-100 discovery metadata for exchange sets to be encoded in the Exchange Catalogue. Figure 12-1 depicts an example of relevant cases. The latest approved S-100 discovery metadata model is located within the latest edition of S-100 Part 4.

S100_DatasetDiscoveryMetadata

+ fileName: CharacterString
+ filePath: CharacterString
+ description: CharacterString
+ dataProtection: Boolean [0..1]
+ protectionScheme: S100_ProtectionScheme [0..1]
+ digitalSignatureReference: S100_DigitalSignature
+ digitalSignatureValue: S100_DigitalSignatureValue
+ copyright: MD_RestrictionCode [0..1]
+ classification: MD_SecurityConstraints [0..1]
+ purpose: CharacterString [0..1]
+ specificUsage: MD_Usage [0..1]
+ editionNumber: CharacterString [0..1]
+ updateNumber: CharacterString [0..1]
+ updateApplicationDate: Date [0..1]
+ issueDate: Date
+ issueTime: Time [0..1]
+ productSpecification: S100_ProductSpecification
+ producingAgency: CI_Responsibility
+ optimumDisplayScale: Integer [0..1]
+ maximumDisplayScale: Integer [0..1]
+ minimumDisplayScale: Integer [0..1]
+ horizontalDatumReference: CharacterString
+ horizontalDatumValue: Integer
+ epoch: CharacterString [0..1]
+ verticalDatum: S100_VerticalAndSoundingDatum [0..1]
+ soundingDatum: S100_VerticalAndSoundingDatum [0..1]
+ dataType: S100_DataFormat
+ dataTypeVersion: CharacterString
+ dataCoverage: S100_DataCoverage [0..*]
+ comment: CharacterString [0..1]
+ layerID: CharacterString [0..*]
+ defaultLocale: PT_Locale
+ otherLocale: PT_Locale [0..*]
+ metadataFileIdentifier: CharacterString
+ metadataPointOfContact: CI_Responsibility
+ metadataDateStamp: Date
+ metadataLanguage: CharacterString [1..*]

S100_ExchangeCatalogue

+ identifier: S100_CatalogueIdentifier
+ contact: S100_CataloguePointofContact
+ productSpecification: S100_ProductSpecification [0..1]
+ metadataLanguage: CharacterString
+ exchangeCatalogueName: CharacterString
+ exchangeCatalogueDescription: CharacterString
+ exchangeCatalogueComment: CharacterString [0..1]
+ compressionFlag: Boolean [0..1]
+ sourceMedia: CharacterString [0..1]
+ replacedData: Boolean [0..1]
+ dataReplacement: CharacterString [0..1]

S100_CatalogueMetadata

+ fileName: CharacterString [1..*]
+ fileLocation: CharacterString [1..*]
+ scope: S100_CatalogueScope [1..*]
+ versionNumber: CharacterString [1..*]
+ issueDate: Date [1..*]
+ productSpecification: S100_ProductSpecification [1..*]
+ digitalSignatureReference: S100_DigitalSignature
+ digitalSignatureValue: S100_DigitalSignatureValue
+ defaultLocale: PT_Locale
+ otherLocale: PT_Locale [0..*]

«enumeration» S100_VerticalAndSoundingDatum

meanLowWaterSprings = 1
meanLowerLowWaterSprings = 2
meanSeaLevel = 3
lowestLowWater = 4
meanLowWater = 5
lowestLowWaterSprings = 6
approximateMeanLowWaterSprings = 7
indianSpringLowWater = 8
lowWaterSprings = 9
approximateLowestAstronomicalTide = 10
nearlyLowestLowWater = 11
meanLowerLowWater = 12
lowWater = 13
approximateMeanLowWater = 14
approximateMeanLowerLowWater = 15
meanHighWater = 16
meanHighWaterSprings = 17
highWater = 18
approximateMeanSeaLevel = 19
highWaterSprings = 20
meanHigherHighWater = 21
equinoctialSpringLowWater = 22
lowestAstronomicalTide = 23
localDatum = 24
internationalGreatLakesDatum1985 = 25
meanWaterLevel = 26
lowerLowWaterLargeTide = 27
higherHighWaterLargeTide = 28
nearlyHighestHighWater = 29
highestAstronomicalTide = 30

S100_SupportFileDiscoveryMetadata

+ fileName: CharacterString
+ fileLocation: CharacterString
+ purpose: S100_SupportFilePurpose
+ editionNumber: CharacterString
+ issueDate: Date
+ supportFileSpecification: S100_SupportFileSpecification
+ dataType: S100_SupportFileFormat
+ otherDataTypeDescription: CharacterString [0..1]
+ comment: CharacterString [0..1]
+ digitalSignatureReference: S100_DigitalSignature [0..1]
+ digitalSignatureValue: S100_DigitalSignatureValue [0..1]
+ defaultLocale: PT_Locale [0..1]

IF S100_SupportFileDiscoveryMetadata.dataType = "ASCII"
1) S100_SupportFileSpecification.name must be one of the MD_CharacterSetCode literals
2) S100_SupportFileSpecification.{version, date} are optional
ELSE
1) S100_SupportFileSpecification.{version, date} are mandatory

S100_DataCoverage

+ ID: Integer
+ boundingBox: EX_GeographicBoundingBox
+ boundingPolygon: EX_BoundingPolygon [1..*]
+ optimumDisplayScale: Integer [0..1]
+ maximumDisplayScale: Integer [0..1]
+ minimumDisplayScale: Integer [0..1]

S100_SupportFileSpecification

+ name: CharacterString
+ version: CharacterString [0..1]
+ date: Date [0..1]

S100_ProductSpecification

+ name: CharacterString
+ version: CharacterString
+ date: Date
+ number: Integer

«enumeration» S100_SupportFilePurpose

new
replacement
deletion

«enumeration» S100_SupportFileFormat

ASCII
JPEG2000
HTML
XML
XSLT
VIDEO
TIFF
PDF/A or U/A
LUA
other

«enumeration» S100_DataFormat

ISO/IEC 8211
GML
HDF5
undefined

«enumeration» S100_ProtectionScheme

S100p154.0.0

«enumeration» S100_DigitalSignature

dsa

S100_DigitalSignatureValue

PT_Locale

+ language: LanguageCode
+ country: CountryCode [0..1]
+ characterEncoding: MD_CharacterSetCode

(ISO 19115-1)

«enumeration» S100_CatalogueScope

featureCatalogue
portrayalCatalogue
interoperabilityCatalogue

S100_CatalogueIdentifier

+ identifier: CharacterString
+ editionNumber: CharacterString
+ date: Date

S100_CataloguePointofContact

+ organization: CharacterString
+ phone: CI_Telephone [0..1]
+ address: CI_Address [0..1]

If additional metadata elements are needed they should be documented in the Product Specification Metadata section and extensions to the generic metadata schemas developed using the standard ISO extension mechanism.

> Prior to the creation of extended metadata, a careful review of the existing metadata within ISO 19115-1 [and the S-100 generic metadata model] must be performed to confirm that suitable metadata does not already exist. If suitable metadata exist within ISO 19115-1 [or the generic S-100 metadata model], then it must be used. (S-100 Part 4a, Appendix 4a-E, extended to include S-100 generic metadata.)

IHO metadata XML schemas for exchange catalogues and discovery metadata have been developed and are available at the IHO software distribution site (https://github.com/IHO-S100WG) and will be served through the IHO GI Registry (http://registry.iho.int) or another IHO site ultimately.

An Exchange Catalogue Builder to be provided by IHO is under development. Exchange catalogues can also be prepared using off-the-shelf commercial and open-source XML editing and authoring tools. Generic tools for ISO metadata can be used for ISO metadata files for each dataset (in S-100 Edition 4 and S-100 Edition 3, the ISO metadata files are separate from, and referenced by, the Exchange Catalogue XML files).

12.1.2   Use of the IHO S-100 generic metadata model and schemas in exchange catalogues

If the generic S-100 Exchange Catalogue format and XML schemas cannot be used as is, either with or without restrictions, then Product Specifications must derive their metadata models from the generic model in S-100 Part 4a (supplemented with Part 4b, for gridded data). Derivations may be restrictions, extensions, or a combination of both. Derivations must conform to the rules in S-100 Part 4a, Appendix 4a-E. Custom UML diagrams similar to Figure 4a-D-4 in S-100 are required for Product Specifications that add classes or attributes, and recommended if Product Specifications omit optional classes/attributes or restrict allowed values in enumerations.

S-100 Part 4a specifies which of the S-100 metadata classes and attributes are mandatory and which are optional. Product Specifications may omit optional S-100 metadata classes or attributes as appropriate.

Restrictions on S-100 classes and attributes must be expressed as constraints. Restrictions include making an optional attribute mandatory or using a subset of attribute values. Restrictions do not require new metadata classes either.

The model can be extended with product-specific classes derived from the classes in the generic model. Derived classes can define additional attributes.

In order to implement a product-specific metadata model, the S-100 XML schemas that encode generic S-100 metadata can be supplemented with:

- Executable constraint checks to apply product-specific restrictions, in Schematron[1] or another language.
- Product-specific schemas that import the generic XML schemas and extend the generic XML types, to capture extensions.

This method means no change to the generic Exchange Catalogue XML schema or XSD files is needed. Instead product-specific customization can be implemented by adding supplementary files to the S-100 generic implementation.

Table 12-1 describes how Product Specifications can describe their metadata models. The actions are elaborations of the allowed extensions listed in S-100 Part 4a, Appendix 4a-E. The UML diagram would be based on S-100 Figure 4a-D-4.

> Product specifications must define their metadata models by either reusing the S-100 generic metadata model or extending the generic model in conformance with the rules in S-100, Appendix 4a-E.
>
> ISO 19115-1 and S-100 Part 4a-E state: 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.
>
> Unnecessary specializations of the generic S-100 metadata classes should therefore be avoided.

---

[1] Schematron rules can be checked using off-the-shelf software like XML editors, but implementations can implement the constraints in any suitable language for production tools or applications which cannot integrate Schematron validation or XSLT in their workflows.

Action	UML diagram	Documentation table	XML implementation
Omit an optional class or attribute	Suppress its display using diagramming tool functionality.	Omit the corresponding table or row.	Rule to check that the XML element <u>is not</u> present.
Make an optional class or attribute mandatory	No change. The multiplicity will still be 0.. but a diagram note may be added stating that it is mandatory in this product.	Multiplicity column should have the mandatory multiplicity and the Remark column a remark stating it is mandatory in this product.	Rule to check that the XML element <u>is</u> present.
Restrict multiplicity	As for the previous row, with appropriate modifications. This is the general case of making an optional attribute mandatory.		
Limit enumeration values to a subset	Suppress display of excluded values using diagramming tool functionality.	Omit the corresponding row.	Rule to check that the value <u>is not</u> used.
Restrict the value of a text, numeric, CharacterString attribute	No change required.	Specify the value in the Remark column.	Rule to check that the value <u>is</u> as specified.
Add a new metadata class or attribute	Specialize the appropriate S-100 metadata class and add the new class or attribute.	Add a new documentation table, or extend the table describing the original class. Inherited attributes should be distinguished from direct attributes.	Additional XSD that imports the generic metadata XML schema and extends its types.
Add a new metadata package in the Exchange Catalogue. (Note: Whether the new package must be in the Exchange Catalogue or the ISO metadata depends on product-specific considerations like its expected use in applications.)	Add the class or classes for the new package and aggregate to a specialization of the Exchange Catalogue container class (S100_ExchangeCatalogue).	New tables for the new classes and extended catalogue container class.	As specified in ISO 19115-3 and S-100 clause 4a-5.6.5). If required in Exchange Catalogue, additional XSD that imports the generic metadata XSD and extends its types.
Add a new metadata package only in ISO metadata.	Diagram(s) extending S-100 Figure 4a-1 and specifying new classes and attributes.	New tables for the new classes.	Describe the extensions in the ISO 19115-3 metadata extension format and extend the ISO metadata XSDs with new types.

Action	UML diagram	Documentation table	XML implementation
Add values to an enumeration. (Consider proposing its addition to S-100 generic metadata via an S-100 maintenance proposal instead of a product-specific extension.)	Specialize the relevant S-100 metadata class and override the affected attribute.	Describe the override and new data type in the appropriate tables.	XML schema override in a derived XSD. Ignore any warnings generated by generic constraint-checking rules. Define new constraint-checking rules if necessary.
Create new metadata codelist elements (expand a codelist.)	Expand codelist in diagram(s). (Some codelists like languages may be too long for a UML diagram.)	Add codelist values to codelist table (if any), or document the new values using another method.	Add new value to appropriate codelist and GML dictionary files.
Create a new metadata codelist to replace the domain of an existing metadata element that has "free text" listed as its domain value. (Note: Simply restricting the character strings as described in an earlier row is a simpler solution.)	Specialize the relevant S-100 metadata class and override the affected attribute. Add a codelist UML element documenting the codelist.	Document the specialization and new codelist in appropriate tables.	XML schema override in a derived XSD. Ignore any warnings generated by generic constraint-checking rules. Define new constraint-checking rules if necessary. Add new codelist to appropriate codelist and GML dictionary files.
No value for a mandatory attribute	None	Specify that the mandatory attribute must be nilled or assigned one of the allowed values as a fixed default.	Rule to check the value.

Table 12-1 - Specifying metadata in Exchange Catalogues for individual Product Specifications

Product Specifications should not clone and rename classes, attributes, and enumerations defined in the generic model, but instead extend the generic model as needed. This requirement helps keep product-specific and generic S-100 metadata harmonized as both S-100 and the Product Specifications evolve over successive versions. Cloning and renaming classes and attributes will result in different branches of metadata. The divergence of metadata branches will increase as both S-100 and Product Specifications evolve over successive versions. One consequence will be a need for implementations to have independent product-specific metadata processing modules. This situation would be undesirable for products that are processed by the same application (such as generic viewers, ECDIS, or ECS).

A typical Product Specification Exchange Catalogue UML diagram is reproduced in Figure 12-2. Compare to Figure 12-1, noting the omission of attributes for vertical and sounding datum from dataset discovery metadata and restrictions of support file format and data format enumerations.
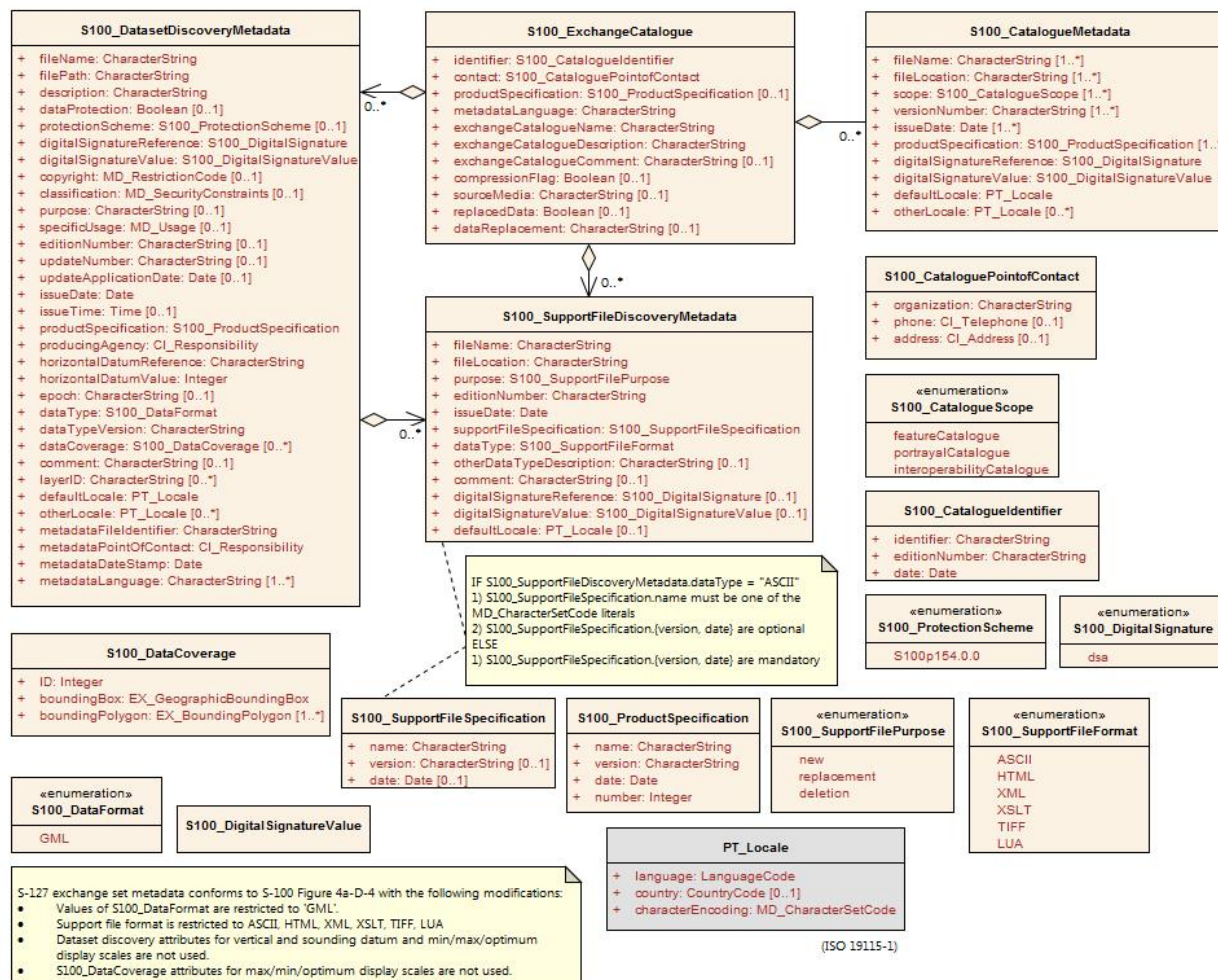
S100_DatasetDiscoveryMetadata
+ fileName: CharacterString
+ filePath: CharacterString
+ description: CharacterString
+ dataProtection: Boolean [0..1]
+ protectionScheme: S100_ProtectionScheme [0..1]
+ digitalSignatureReference: S100_DigitalSignature
+ digitalSignatureValue: S100_DigitalSignatureValue
+ copyright: MD_RestrictionCode [0..1]
+ classification: MD_SecurityConstraints [0..1]
+ purpose: CharacterString [0..1]
+ specificUsage: MD_Usage [0..1]
+ editionNumber: CharacterString [0..1]
+ updateNumber: CharacterString [0..1]
+ updateApplicationDate: Date [0..1]
+ issueDate: Date
+ issueTime: Time [0..1]
+ productSpecification: S100_ProductSpecification
+ producingAgency: CI_Responsibility
+ horizontalDatumReference: CharacterString
+ horizontalDatumValue: Integer
+ epoch: CharacterString [0..1]
+ dataType: S100_DataFormat
+ dataTypeVersion: CharacterString
+ dataCoverage: S100_DataCoverage [0..*]
+ comment: CharacterString [0..1]
+ layerID: CharacterString [0..*]
+ defaultLocale: PT_Locale
+ otherLocale: PT_Locale [0..*]
+ metadataFileIdentifier: CharacterString
+ metadataPointOfContact: CI_Responsibility
+ metadataDateStamp: Date
+ metadataLanguage: CharacterString [1..*]

S100_ExchangeCatalogue
+ identifier: S100_CatalogueIdentifier
+ contact: S100_CataloguePointofContact
+ productSpecification: S100_ProductSpecification [0..1]
+ metadataLanguage: CharacterString
+ exchangeCatalogueName: CharacterString
+ exchangeCatalogueDescription: CharacterString
+ exchangeCatalogueComment: CharacterString [0..1]
+ compressionFlag: Boolean [0..1]
+ sourceMedia: CharacterString [0..1]
+ replacedData: Boolean [0..1]
+ dataReplacement: CharacterString [0..1]

S100_CatalogueMetadata
+ fileName: CharacterString [1..*]
+ fileLocation: CharacterString [1..*]
+ scope: S100_CatalogueScope [1..*]
+ versionNumber: CharacterString [1..*]
+ issueDate: Date [1..*]
+ productSpecification: S100_ProductSpecification [1..*]
+ digitalSignatureReference: S100_DigitalSignature
+ digitalSignatureValue: S100_DigitalSignatureValue
+ defaultLocale: PT_Locale
+ otherLocale: PT_Locale [0..*]

S100_SupportFileDiscoveryMetadata
+ fileName: CharacterString
+ fileLocation: CharacterString
+ purpose: S100_SupportFilePurpose
+ editionNumber: CharacterString
+ issueDate: Date
+ supportFileSpecification: S100_SupportFileSpecification
+ dataType: S100_SupportFileFormat
+ otherDataTypeDescription: CharacterString [0..1]
+ comment: CharacterString [0..1]
+ digitalSignatureReference: S100_DigitalSignature [0..1]
+ digitalSignatureValue: S100_DigitalSignatureValue [0..1]
+ defaultLocale: PT_Locale [0..1]

S100_CataloguePointofContact
+ organization: CharacterString
+ phone: CI_Telephone [0..1]
+ address: CI_Address [0..1]

«enumeration» **S100_CatalogueScope**
featureCatalogue
portrayalCatalogue
interoperabilityCatalogue

S100_CatalogueIdentifier
+ identifier: CharacterString
+ editionNumber: CharacterString
+ date: Date

IF S100_SupportFileDiscoveryMetadata.dataType = "ASCII"
1) S100_SupportFileSpecification.name must be one of the MD_CharacterSetCode literals
2) S100_SupportFileSpecification.{version, date} are optional
ELSE
1) S100_SupportFileSpecification.{version, date} are mandatory

«enumeration» **S100_ProtectionScheme**
S100p154.0.0

«enumeration» **S100_DigitalSignature**
dsa

S100_DataCoverage
+ ID: Integer
+ boundingBox: EX_GeographicBoundingBox
+ boundingPolygon: EX_BoundingPolygon [1..*]

S100_SupportFileSpecification
+ name: CharacterString
+ version: CharacterString [0..1]
+ date: Date [0..1]

S100_ProductSpecification
+ name: CharacterString
+ version: CharacterString [0..1]
+ date: Date
+ number: Integer

«enumeration» **S100_SupportFilePurpose**
new
replacement
deletion

«enumeration» **S100_SupportFileFormat**
ASCII
HTML
XML
XSLT
TIFF
LUA

«enumeration» **S100_DataFormat**
GML

S100_DigitalSignatureValue

PT_Locale
+ language: LanguageCode
+ country: CountryCode [0..1]
+ characterEncoding: MD_CharacterSetCode

(ISO 19115-1)

S-127 exchange set metadata conforms to S-100 Figure 4a-D-4 with the following modifications:
- Values of S100_DataFormat are restricted to 'GML'.
- Support file format is restricted to ASCII, HTML, XML, XSLT, TIFF, LUA
- Dataset discovery attributes for vertical and sounding datum and min/max/optimum display scales are not used.
- S100_DataCoverage attributes for max/min/optimum display scales are not used.

Figure 12-1. Example Exchange Catalogue metadata classes for S-127 (Marine Traffic Management)

12.1.3 Extensions in ISO metadata files

S-100 Part 4a, clause 4a-5.6.5 specifies how extensions must be documented. This specification is derived from the ISO Specification.

12.1.4 Data protection, authentication, and encryption

The purpose of data protection in S-100 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 for which a customer has acquired a license.
3) Authentication: To provide assurance that the products were actually created and distributed by the producers and distributors who are identified as such in the product package or datasets.

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. Authentication is provided by means of digital signatures applied to the product files. Selective access to individual products is supported by providing users with a licensed set of data permits. The 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.

12.1.4.1 Application of protection measures

Product Specifications should specify whether datasets must, must not, or may be encrypted using the S-100 recommended security scheme using the *dataProtection* attribute of **S100_DatasetDiscoveryMetadata**. If this Boolean attribute is set to TRUE the co-attribute *protectionScheme* must also be assigned a value from the enumeration of security schemes (**S100_ProtectionScheme**). S-100 at present defines only one recommended security scheme, which is described in Part 15.

12.1.4.2 Digital signatures

S-100 Edition 4.0.0, Part 4a made digital signatures mandatory for datasets and catalogues in exchange sets. For support files, digital signatures are allowed but optional. Product Specification authors should therefore specify which support files must, must not, or may be signed. Signing datasets is independent of whether the recommended security scheme is applied to the dataset.

The digital signature method is encoded in attribute *digitalSignatureReference*. There is also an attribute *digitalSignature* for encoding the digital signature itself. These attributes are present in dataset discovery metadata, catalogue metadata, and support file discovery metadata (classes **S100_DatasetDiscoveryMetadata**, **S100_ExchangeCatalogue**, and **S100_SupportFileDiscoveryMetadata**). The structure of the digital signature is specified in S-100 Part 15. S-100 Edition 4.0.0 Part 4a is quite permissive as to the format, allowing either well-formed XML (one or more XML elements from an unspecified schema) or a character string (for example base64 encoded signature).

12.1.4.3 Compression and archive format

Compression of data products as used in S-100 Parts 4a and 15 includes specification of the archive format as well as actual file compression method. In S-100 Edition 4.0.0 there is only one archive format (ZIP) and only one allowed compression method (DEFLATE). Compression requires packing into an archive. Product Specification authors must specify whether an exchange set must, must not, or may be compressed by specifying appropriate constraints on the *compressionFlag* metadata attribute in exchange catalogues. Further information about archive format and compression method is provided in S-100 Part 15, (clause 15.5). S-100 Edition 4.0.0 provides for exchange catalogues to have only one instance each of the *compressionFlag* attribute, which therefore applies to all files in the exchange set (that is, after compression, there will be only one ZIP archive which contains all data files, support files, and catalogues in the exchange set, with the DEFLATE compression method applied to all[2]).

Product Specification authors should note that that an exchange set may contain other exchange sets. Each included exchange set can be treated as an individual for compression purposes (that is, packed into a ZIP archive or not), but they will all be packed into the archive of the overall container exchange set, either as a folder hierarchy or single-file ZIP archives, depending on whether they are individually compressed.

The encryption and digital signature features of ZIP are not used.

---

[2] Some Zip implementations may check whether the compression method actually produces a smaller file, and if not, store the original instead. It may be possible to force compression even if the file size is not reduced.

12.1.5 Metadata for imagery and gridded data

Exchange set metadata for Product Specifications dealing with imagery and gridded information is the same as for ordinary vector datasets (clauses 12.1.1 – 12.1.4) extended with additional metadata elements and attributes specific to imagery or gridded data. S-100 Part 4b describes the additional metadata elements, which are defined in detail in ISO 19115-2.

The Exchange Catalogue for such products will be as for vector data products. As for vector data, the Product Specification team may define additional product-specific metadata elements and attributes in conformance with S-100 Appendix 4a-E. ISO-defined elements and attributes should be in the ISO metadata file.

12.1.6 Embedded metadata

Certain metadata may be embedded in dataset headers (ISO 8211 and GML formats – S-100 Parts 10a and 10b) or defined attributes and groups (HDF5 format – Part 10c). While specifying embedded metadata is up to individual Product Specifications, this guideline recommends that only metadata considered essential to identifying and reading the dataset should be embedded. Examples are dataset name, persistent global identifier, and MRN (if any), as well as bounding box, number of objects of each type, and CRS identification. Gridded data products may encode spatial representation information such as grid spacing and grid bounding box as metadata for individual features. Gridded data products may also encode structural metadata such as a code indicating the type of grid (regular, irregular, etc.).

12.2   Metadata for services

This section describes metadata for products that are delivered as services.

12.2.1 Generic metadata for services

Generic metadata for services is a work in progress as of August 2018. S-100 Part 4a defines a metadata model for services which conforms to the ISO 19115-1/2 model. S-100 Part 14 defines service metadata differently (**S100_ServiceMetaData** – clause 14-8.1.1). It is left to Product Specifications to determine if servers need to supply service identification metadata in the sense of the underlying ISO standards (reproduced in S-100 Figure 4a-A-2).

Product Specifications needing to define service metadata (identifying and describing the service itself) should use or extend the **SV_ServiceIdentification** class in S-100 Figure 4a-2. Any extensions or restrictions should conform to the ISO rules as described in S-100 Part 4a Appendix 4a-E.

12.2.2 Use of the IHO S-100 metadata model and schemas

S-100 Part 14, clause 14-9.1.3) specifies a *GetMetaData()* operation with *CharacterString* return parameter *exchangeSet* defined as "The exchange set describing the datasets." It is not clear at this time whether this is the metadata defined in S-100 Part 14 clause 14-8 or Exchange Catalogue of S-100 Part 4a, Figure 4a-D-4. This can be considered **payload metadata** (metadata describing the transferred information) to distinguish it from metadata that describes the service itself.

If Product Specifications need to define a subset and/or extension of the S-100 Exchange Catalogue to describe payload metadata, they should use the same methods specified in S-100 Part 4a/4b and clause 8.1 of this Guidance. Note that these sections require conformance to the ISO rules described in S-100 Part 4a, Appendix 4a-E.

12.2.3 Data protection, authentication, and encryption

Many of the data protection, authentication, and encryption provisions of S-100 Part 15 should also apply to services, but details and implementation constraints will differ – for example, a data product may create its transfer package using a format other than ZIP, and the connection protocol may need to be secure (for example HTTPS rather than HTTP). Product Specification teams which need to use security schemes in service transfer modes are encouraged to explore using or adapting S-100 Part 15 and report on their experiences.

12.2.4 Embedded metadata

The considerations are similar as for embedded metadata described in clause 12.1.6. Additional considerations may arise from the transfer mode; for example, giving greater importance to minimizing data volume. Product Specification teams are encouraged to consider using or adapting the embedded metadata principles in clause 12.1.6 and the methods and constraints of the data format which is used (ISO 8211, GML, HDF5, or other format) and report on their experiences.

13 Define data encoding format

13.1 Selection of encoding format

The encoding format should be selected based on the type of product and other requirements, including production and processing. The characteristics of the three standard data formats included in S-100 Edition 4.0.0 are summarized below for convenience.

	ISO 8211	GML	HDF5
Type of product to which suited	Nautical charts and feature-heavy vector data	Nautical publications and information-heavy vector data; discrete weather information; small datasets such as marine safety information; data delivered via messages and web services.	Coverage-based data
Generic data format	Yes	Yes	Yes
Data production complexity	Requires custom tools	Can be produced with a range of tools from text editors to custom apps and database SQL queries.	Custom apps that use off-the-shelf libraries
Processing complexity	High	Low	High
Supporting off-the-shelf software	Yes	Off-the-shelf viewers and server software; can be viewed with ordinary text editors.	Off-the-shelf viewer
Data volumes	Lower	Higher	Lower
Type of data	Vector	Vector; coverage schemas are defined in	Gridded

	ISO 8211	GML	HDF5
		the GML specification but not used in S-100.	
Supporting artifacts needed in Product Specifications	Feature Catalogue	XML schemas for data validation; datasets can be processed by apps without XML schemas; self-documented format (tags indicate objects and attributes); Feature Catalogue optional.	Embedded object and attribute tags; Feature Catalogue optional.

Table 133-13-1 - Characteristics of S-100 standard encoding formats

While other formats than the three standard encodings are possible, the use of a non-standard format has the following implications:

- Loss of genericity, requiring special purpose development by implementers or conversion to a standard format with impacts on performance.

- Potential loss of compatibility with interoperability.

The question of appropriate formats for transactional, web-service, or message-based information has not been formally addressed in depth at this time, but Product Specifications needing such delivery modes should endeavour to use one of the standard formats in order to minimize implementation complexities.

13.2 Data format definition artefacts for GML

Selection of the GML format will require definition of XSD files encapsulating the S-100 Application Schema as XSD files conforming to the GML specification (ISO 19136 and S-100 Part 10b). There must be a 1/1 mapping from the Application Schema to XML schema constructs in the GML XSD files that implement the model as GML Application Schemas.

- Feature and information classes in the UML domain model map to XML complex types and XML elements. The class (camel case) name is the same as the element name (XML tag excluding any namespace).

- Simple attributes in the UML domain model map to XML elements contained in the feature and information complex types mentioned above. The names (XML tags, excluding any namespace) are the same as the (camel case) names of the UML attributes.

- Complex attributes in the UML domain model map to XML elements contained in the feature and information complex types mentioned in the first bullet. The names (XML tags, excluding any namespace) are the same as the (camel case) names of the UML attributes. The types of these XML elements are XML complex types with the same (camel case) name, either with or without "Type" suffixed.

- Enumerations map to XML enumerations of the same (camel case) name.

- Codelists map to XML union types of the same (camel case) name; one member of the union type is an anonymous enumeration of the listed values and another member is a string restricted to the pattern "other: ..." as specified in S-100 Part 3, clause 3-6.7.

- Associations are encoded as elements contained in the feature and information complex types, named according to the role indicated in the UML domain model and with a GML attribute that contains the gml:id of the referenced object.

Schema developers should note that conformance to the GML specification requires conformance to the rules set forth in the GML specification (ISO 19136 / OGC 07-036), not merely XML-validation against the GML schemas.

S-100 Part 10b describes the S-100 GML profile. The XSD files for the S-100 GML profile are available at the IHO S-100 distribution site (https://github.com/IHO-S100WG). Previously defined GML schemas for other Product Specifications (for example S-122, S-123) will provide useful guidance and will be made available through the IHO Product Specification distribution site or GitHub distribution site.

Guidance for generic processing for GML datasets is included in S-100 Edition 4.0.0.

14 Portrayal elements and rules

Portrayal Catalogues are necessary only for data products that are intended to be displayed graphically (as opposed to text or other processing).

Often there will be only one intended application domain – generally ECDIS – and a limited number of closely related usage scenarios (that is, route monitoring/planning and voyage planning) utilizing one Portrayal Catalogue. Some data products may not be intended for display, in which case no Portrayal Catalogue need be prepared. Others may be intended for more than one application domain, in which case the possible need for different Portrayal Catalogues should be considered. Different usage scenarios may also benefit from different Portrayal Catalogues.

S-100 defines an XSLT-based portrayal mechanism in Part 9. Edition 4.0.0 adds the LUA scripting language as a second portrayal mechanism and defines a standard mechanism for including scripting support in S-100 based products. Developers/OEMs may prefer LUA due to promises of performance and similarity to common programming languages.

The portrayal section of a Product Specification should include:
- Pictorial representations of symbols and colours, accompanied by symbol specifications (the latter preferably in the form of machine-processable files as well as formal specifications in the text of the portrayal section).
- Specifications and recommendations for the use of symbols by implementers, e.g., calculating orientation from attribute values and the use of thinning to reduce crowding of displays at small display scales, display of text accompanying symbols, masking of boundaries, etc.
- Products that need to be included in ECDIS must have day, night, and dusk palettes defined.

15 Registration of portrayal elements

Portrayal elements must be registered in the GI Registry Portrayal Register. Elements that need to be registered are listed below. A portrayal element needs to be registered only if it is new[3] to the Register – elements that are already in the Register need not be registered again.
- New symbols, fonts, line styles, area fills, and pixmaps.
- New display modes
- New viewing group layers
- Viewing groups

---

[3] This clearly introduces a dependency between different data products. The Portrayal Register design and business processes to manage such dependencies need to be worked out. This section may have to be revised after that is done.

- New display planes
- New colour tokens
- New colour profiles used by portrayal of this product. If there are multiple palettes (for example day/night/dusk), the colour profile file(s) must define all palettes.
- New CSS files encoding the colour tokens and RGB values used by all symbols in the portrayal of this product
- New context parameters.
- New display priorities.

16 Portrayal Catalogue

An IHO Portrayal Catalogue Builder is available to product specification developers.

17 Reference systems

The preferred coordinate reference system is EPSG 4326 which is based on the WGS 84 horizontal datum.

Horizontal datum will normally be referenced by giving its code in the EPSG Register. If the coordinate reference system is not one of the coordinate reference systems in the EPSG Register, a datum may be specified in a support file as described in S-100 Part 6, should there be a use case in the scope of the data product.

A set of vertical datums is listed in S-100 Appendix 4a-D (S100_VerticalAndSoundingDatum). Product Specification developers are encouraged to adopt the S-101 ENC datum (either standard or local S-101 ENC datum) as a common vertical reference datum if possible. If a need for an additional datum is identified, it should be proposed as a revision to S-100.

S-100 includes 'local datum' as an allowed value for datum attributes, but this is of limited utility even within a data product. (S-101 cites an example of use of local datum in a non-tidal basin, where depths may refer to a sounding datum different from that in open waters. If this area is navigable at the maximum display scale of the ENC data, the value of this datum must be encoded using attribute vertical datum = 24 (local datum), in a meta-feature co-incident with the area covered by the dock.) Data conversion to/from unspecified local datums would be problematic.

18 Data product delivery

18.1   Delivery content and structure

Define the content and structure of delivery packages: Exchange sets, messages, or web services.

18.1.1 Exchange sets

Exchange set structure should be defined, either by using the structural diagram from S-100 Appendix 4a-D (reproduced below) as is, or restricting the allowed components, or defining extensions of the individual components. If there is internal structure in the exchange sets (for example folders and sub-folders), determine the required layout and naming conventions. Determine how the exchange set as a whole is packaged (for example ordinary folders, zip file, etc.).
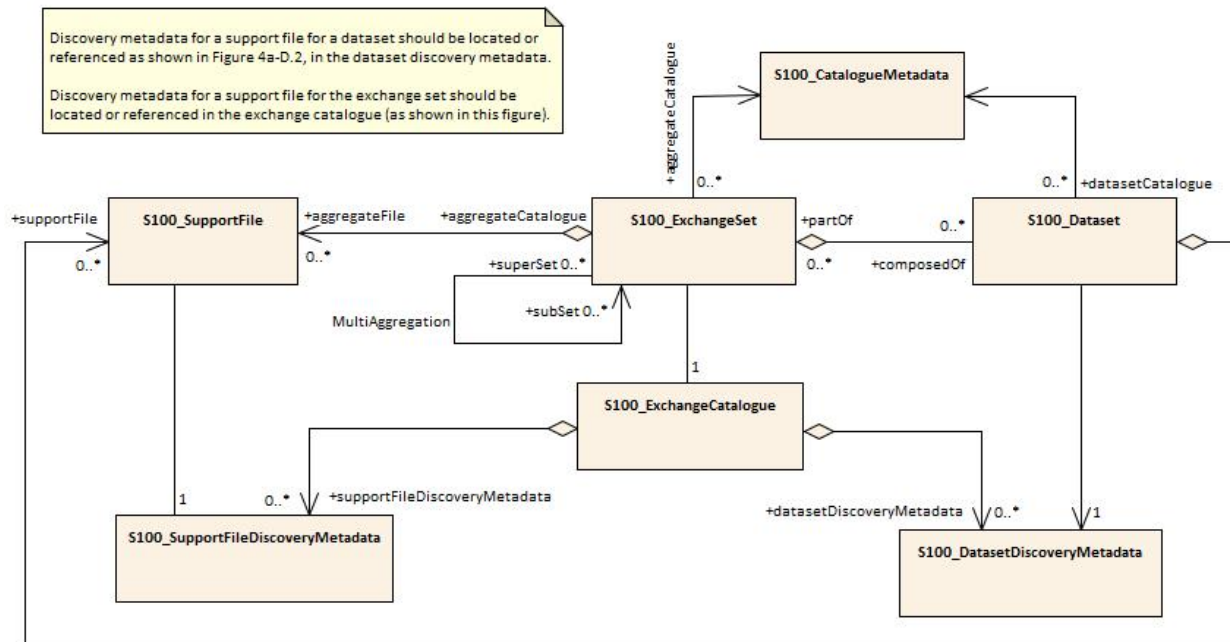
Figure 18-1 - Prototype exchange set structure

Determine what naming conventions, if any, must apply to individual components of the exchange set - dataset files, exchange sets, and support files. Naming conventions are generally not needed for message-based or service modes of data delivery, but a unique identifier will generally be needed for each message or service transaction.

18.1.2 Services

S-100 Part 14 describes three types of communication:

- Session oriented communication (S-100 Part 14, clause 14-4) - point-to-point connections between client and server described by an interaction model that describes the life span of a session (initiation, maintenance and termination of the session).

- Session-less interactive communication (S-100 Part 14, clause 14-5) - an encapsulation of all relevant information within a request, based on which the server formulates an appropriate response.

- Message streams (S-100 Part 14, clause 14-6) - a unidirectional flow of messages containing well-defined sets of data.

For message and web service modes of delivery, specify the container format, packaging, and specify the transfer protocol (for example REST (Representational state transfer), SOAP (Simple Object Access Protocol)) and packaging (for example WSDL (Web Services Delivery Language), WFS (Web Feature Service)). S-100 Part 14 currently permits three service technologies: SOAP, REST, and CORBA.

[Developing specifications for online services will be included in a later version of this document, pending more experience with S-100 Part 14.]

18.2   Dataset updates

18.2.1 General considerations for updates

Define the conditions and mechanisms for data updates, specifically:

- Update cycles – how frequently data must be updated, whether updates are issued on a regular cycle, as-needed, or a combination of both;
- How long each dataset is valid and how to validity periods will be indicated;
- Mechanisms for cancellation, replacement, and reissue of datasets;
- Metadata for updates;
- Types of updates – whether the data product requires incremental, whole-dataset replacement, irregular, or cumulative updates;
- Criteria for determining when datasets must be replaced by new datasets, superseded, reissued, updated, and cancelled.

18.2.2 Format-specific update considerations

ISO 8211: S-100 Part 10a describes update mechanisms for ISO 8211 datasets. Each Product Specification must define the structure of update datasets.

GML: Replacement of whole objects is the recommended method, but there are XML specifications that allow update of individual attributes. (Details will be included in a later version of this document, pending further discussions.)

HDF5: Feature (coverage) can be updated in its entirety or in part (the update can be a sub-grid).

18.3   Supporting information

Describe how any auxiliary content is delivered either with or as an adjunct to data. S-100 provides for 'support files' to be included in exchange sets. Support files can be graphic or text information files referenced by dataset objects, or other files such as dictionaries and catalogues (including Feature or Portrayal Catalogues). Define allowed file formats and naming conventions for support files.

Note that since Feature and Portrayal Catalogues are shared by all datasets conforming to a specific version of a Product Specification, it will generally be more efficient to deliver Feature and Portrayal Catalogues once rather than with every exchange set. Possible methods of such special deliveries have not been standardized yet and are left to Product Specification developers, but may include special exchange sets distributed through the usual channels or a centralized means such as publication on a web server. If a central distribution mechanism is adopted consideration must be given to the possible needs of end users who have only infrequent or no access to the distribution hub (for example low-bandwidth or no Internet access).

19 Validation checks and data quality

At least two types of validation checks are needed:

- Dataset validation checks, for individual datasets. These checks operate on individual objects in datasets and on individual datasets as a whole. They should check the integrity of individual objects in the dataset (spatial, feature, and information types); associations between objects in the

dataset; any embedded metadata or header information in the dataset; and support files referenced in the dataset.

- Package validation checks, for verifying the structure and content of packages (for example exchange sets) and accompanying metadata.

A common set of validation checks is under development (see S-97 Part C). A recommended set of data quality measures has been developed based on dataset statistics derived from the validation checks. The recommended measures are also described in S-97 Part C.

19.1 Validation checks for datasets

Validation checks for datasets should cover:

1) Completeness, including population of attributes and presence of required information, complex attributes without sub-attributes, etc.
2) Logical consistency; for example missing association targets.
3) Spatial consistency; for example topological sanity checks for non-crossing external boundaries, excessive vertex density in lines, etc.
4) Positional accuracy; for example closeness of reported coordinate values to accepted or known absolute or relative coordinate values.
5) Temporal accuracy; for example closeness of reported time measurements to accepted or known values accepted as or known to be true, correctness of the order of events, or validity of data with respect to time.
6) Thematic accuracy, such as attribute values that are consistent with any other related attributes and within allowed ranges or sets.
7) References to support files.
8) Other requirements specific to the product; for example encryption, signatures, etc.

Some of these issues are addressed in the common set of validation checks. A recommended common set of validation checks is described in S-97 Part C. Product Specification developers should supplement the common set with such additional checks as are appropriate for the particular products.

19.2 Validation checks for packages

Validation checks for packages should cover:

1) Package completeness – whether all required components are included, including datasets, support files, metadata, and appropriate catalogues (for example exchange set catalogues, Feature Catalogues, and Portrayal Catalogues). Note that the Product Specification must indicate which catalogues are appropriate to the delivery method – for example, message-based delivery methods may not include catalogues in the delivery packages.
2) Package container format and structure – whether the package is in the approved container format (for example ISO 8211, TIFF, etc) and whether appropriate encryption and signatures have been applied at the container level. Examples of package validation checks are:

- Assuming the Product Specification specified delivery as zip files, is the container a zip file of the appropriate type?
- If the package is arranged in a directory (folder) structure, are the structure and names of directories (folders) as required in the Product Specification?

Package validation checks are required to validate delivery packages, but are expected to be out of the scope of S-97 Part C and Product Specification teams may have to specify their own. The tests for exchange set and service delivery modes will obviously be different, but the matter of validating the delivery package or stream should be addressed for all delivery modes covered in the Product Specification.

19.3   Common validation checks

Given that some features, information types, and Application Schema constructs are used in multiple products, there will be validation checks in common with existing Product Specifications and any such related Product Specifications should be consulted for validation checks. Spatial consistency checks in particular, as well as consistency checks related to meta-features, can be expected to be in common with several data specifications.

Spatial operations used in validation checks must be the operations defined in IHO ENC Validation Checks (S-58 Edition 6.0.0 or its successor).

19.4   Validation checks for base versus update datasets

If the Product Specification defines an update dataset format, the validation checks developed for new (base) datasets should be reviewed for their applicability to update dataset formats.

20 Preparing for interoperability

Product Specification developers should carry out the activities described below if the data product is a candidate for inclusion in an Interoperability Catalogue (S-98). Further details on these activities are provided in S-98, especially S-98 sections 8 and 10.

- Determine which if any product groups in Interoperability Catalogues are supplemented or enhanced by the data product.

- Determine whether and how the IHO Interoperability Catalogue will be affected by the new product, including updates to display priorities, interleaving, predefined combinations, and other interoperability rules and operations.

- Revise Portrayal Catalogues upon recommendation by the IHO Interoperability Team.

- Compare the Application Schema to related Product Specifications and advise the developers of the Interoperability Catalogue on their similarity to features in other Product Specifications. This comparison should include feature concept similarity, attribute bindings, attribute value domains, potential discrepancies in feature geometries, etc.

- Examine the Specification scope and factors affecting the quality of information which may also be present in other products - for example, whether information is for context or background in one product but a significant part of the other. The results of this analysis should be communicated to Interoperability Catalogue developers.

- Consider interactions due to cartographic objects; for example text placement issues in interoperability mode. Product Specification teams should consult with OEMs on how to mitigate such issues.

- Advise Interoperability Catalogue developers on whether features are supplementary information for features in another product. Such features may be combined as part of interoperability.

- Attempt to harmonize effects of maximum and minimum display scales used by datasets in order to avoid situations where one data product is within display scale but a complementary product is not.

21 Sample data / test datasets

Test data should be created in sufficient quantity to validate the main characteristics of the Application Schema. Specifically, the first test dataset should contain:

- At least one instance of each kind of feature and information type.
- A representative set of feature and information associations, preferably at least one instance of each named association. (It is not necessary to create an instance for each and every pair of object classes which may be linked by an association.)
- At least one instance of each meta-feature and data quality feature.

At least one update dataset should also be prepared, to validate the update dataset format and packaging.

Additional test datasets should test typical data volume, representative data capture problems, and error cases.

If delivery is intended to be in the form of exchange sets, the test datasets should be packaged as complete exchange sets, including sample metadata files. Sample packages for other forms of delivery (transactions, messages, web services, etc.) should be emulated as realistically as is practical at this stage – that is, setting up a web server, service broker, etc for web services should be done if doing so is practical but is not an essential requirement (it can and should be done as part of the testing stage).

22 Testing and feedback

A formal test plan should be prepared, including test cases.

23 Work processes

23.1   Registration and requesting an S- number

Apply for a Product Specification number – this should ordinarily be assigned when the development project is approved during the initiation stage.

Registration of Product Specification artefacts in the IHO GI registry should be done in accordance with the procedures established in IHO publication S-99.

23.2   Project Teams

Project Teams should involve domain specialists, information modelling specialists, and representatives of OEM/developer communities.

23.3   Iterative refinement as a development process

Development should plan for iterative refinement, with the following being reviewed at the indicated stages of development:

- Initial Application Schema. Reviews of subsequent revisions can be rolled into the reviews of the main Product Specification document;
- First drafts of main document of Product Specification and DCEG. Subsequent revisions should be reviewed as ready;
- Feature Catalogue, data format and sample datasets should be checked after each major revision to the Application Schema and Feature Catalogue;

- Portrayal Catalogue – first draft and significant revisions;
- Other artefacts or components, such as validation tests – when substantially completed, and after revisions due to changes to the Application Schema, Feature Catalogue, or data format;
- The Product Specification as a whole – after the complete package is ready.

Reviews of different components can be combined to fit the development schedule or Working Group meeting schedules. The Application Schema, main Product Specification, and DCEG should be expected to undergo multiple reviews during different stages of development.

Reviews during the development stage should be requested from:

- Project Team – initial reviews;
- Technical group(s) sponsoring the Specification, as well as related technical Working Groups – after some stability has been achieved;
- Developers, implementers, and OEMs – formal reviews after initial reviews and stabilization in the Project Team and sponsoring technical Working Group. Note that individual implementer/OEM/developers should ideally be involved from the earliest stages of development if available.

Reviews should be completeness, correctness, ability to capture and express the domain, performance/efficiency, as well as conformance reviews for verifying compliance to S-100 and underlying Standards.

In addition, the Data Quality Working Group should review the Product Specification for completeness.

Stakeholder reviews should be requested as the Specification matures, and should involve:

- Producers;
- Developers and OEMs.

Subsequent stages should involve users and user testing; and preparation of an impact study.

Test development and testing should commence upon the Feature Catalogue and data format achieving reasonable stability, presumably after one or two cycles of review.

Final assessment will be at the HSSC level for IHO or equivalent for other organizations.

Pre-publication review will be conducted by the IHO or other publishing organization prior to publication, to check production issues.

23.4   Maintenance of Product Specifications

Clarifications, corrections, and revisions should be designated in accordance with the same criteria used for S-100 as described in S-100 Part 12, Clause 12-2 (Maintenance Procedures).

Product Specifications should undergo periodic review. A two-year review period is suggested for new Specifications, which may be increased to five years after the Specification reaches maturity.

Page intentionally left blank