TSMAD27-4.3.1D

S-100 Part 3

General Feature Model and Rules for Application Schema

3-1 Scope

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

The scope of this Part includes:

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

The following is outside scope:

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

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

3-2 Conformance

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

1) A new S100_GF_NamedType is introduced.
2) A new S100_GF_ObjectType is introduced as a specialisation of S100_GF_NamedType.
3) A new S100_GF_InformationType is introduced as a specialisation of S100_GF_ObjectType, it is constrained to associations with S100_GF_ThematicAttributeType.
4) S100_GF_FeatureType is a specialization of S100_GF_ObjectType,
5) S100_GF_AttributeType is a specialization of GF_AttributeType in that it is abstract in S-100.
6) A new abstract S100_GF_SimpleAttributeType is introduced as a specialisation of S100_GF_ThematicAttributeType.
7) GF_Operation is not used.
8) GF_InheritanceRelation is not used, feature inheritance is represented by the association inheritance.
9) The association attributeOfAttribute is not used. The concept of the complex attribute is used in S-100 to perform a similar function.
10) S100_GF_AssociationType does not use the generalization association between GF_AssociationType and GF_FeatureType. Instead is it is a specialisation of S100_GF_NamedType.
11) S100_GF_AssociationType is associated with S100_GF_ThematicAttributeType by a UML aggregation relationship. This means associations can have descriptive characteristics.
12) New metaclasses S100_GF_FeatureAssociationType and S100_GF_InformationAssociationType are introduced as specialisations of S100_GF_AssociationType.
13) The association role linkBetween of the GF_FeatureType/GF_AssociationType relationship in ISO 19109 is realized as follows:

a) role linkBetween of the S100_FeatureType/S100_GF_FeatureAssociationType relationship
b) role linkBetween of the S100_InformationType / S100_GF_InformationAssociationType relationship
c) role informationLink of the S100_ObjectType / S100_InformationAssociationType relationship.

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

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

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

3-3 References

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

ISO 19106:2003 Geographic information - Geographic Information - Profiles

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

ISO 19107:2003 Geographic information - Spatial schema

ISO 19109:2005 Geographic information - Rules for application schema

ISO 19110:2005 Geographic information - Methodology for feature cataloguing

ISO 19115:2005 Geographic information - Metadata

ISO/CD 19115-2 (N1931) Geographic information - Metadata - part 2

3-4 Context

3-4.1 Objects

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

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

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

3-4.2 Derivation of the General Feature Model

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

3-5 Principles for defining features and information types

3-5.1 Identifiable objects

3-5.1.1 Features

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

3-5.1.2 Information types

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

EXAMPLE A chart note may be modelled as an information type

3-5.2 The General Feature Model

3-5.2.1 Introduction

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

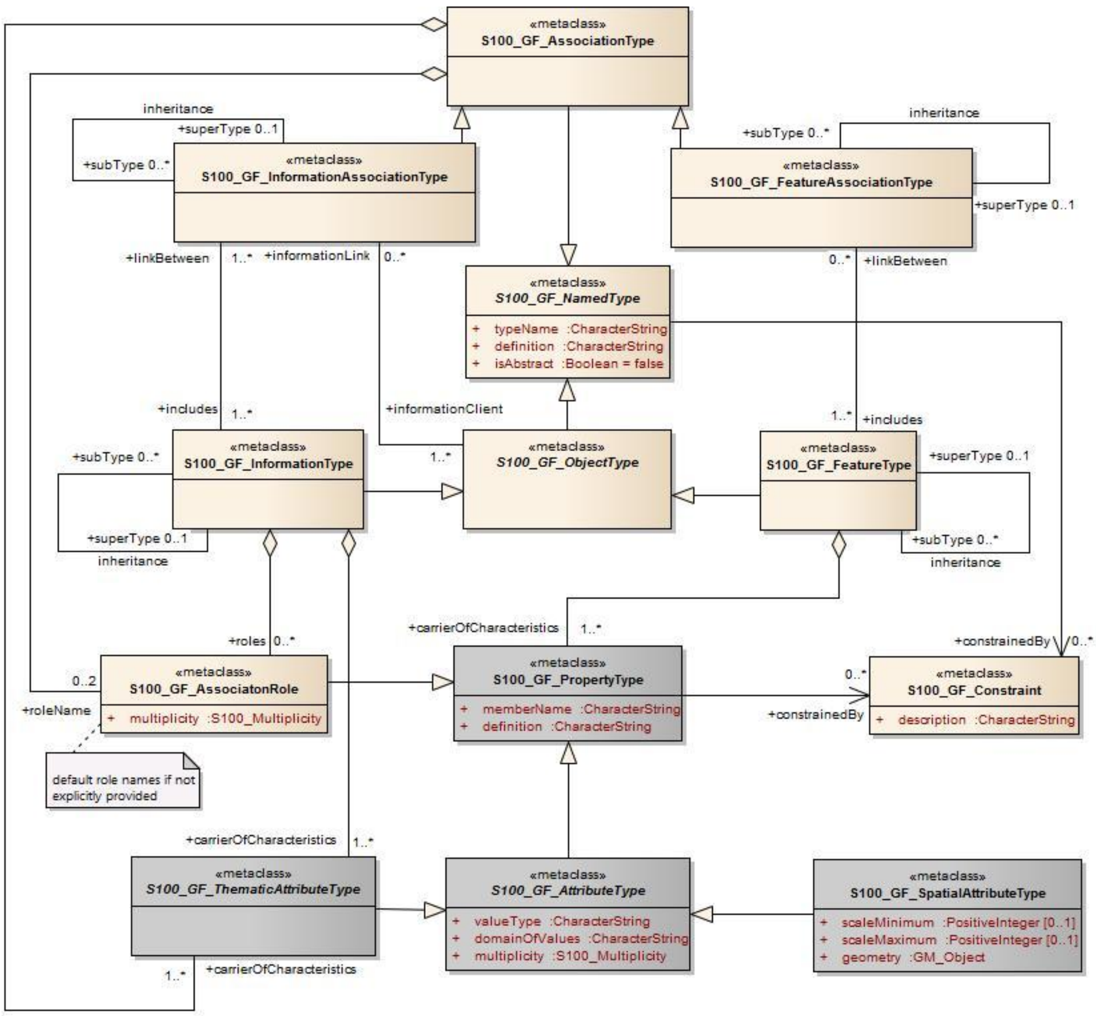


Figure 3-1 — The General Feature Model

3-5.2.2 The purpose of the GFM

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

3-5.2.3 The main structure of the GFM

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

The following clauses define the elements of the GFM.

3-5.2.4 S100_GF_NamedType

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

Table 3-1— S100_GF_NamedType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_NamedType	Abstract base class for object types and association types within the GFM.	-	-
Attribute	typeName	Name of the named type. The name shall be unique within a namespace.	1	CharacterString
Attribute	definition	Definition that describes the named type.	1	CharacterString

Attribute	isAbstract	If true, the named type acts as an abstract supertype. It is not possible to create an instance of an abstract type	1	Boolean
Role	constrainedBy	The role specifies that a constraint is made on the named type.	0..*	S100_GF_Constraint

3-5.2.5 S100_GF_ObjectType

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

Table 3-2— S100_GF_ObjectType

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

3-5.2.6 S100_GF_FeatureType

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

It is a sub-type of the class S100_GF_NamedType;

It does not realise the Generalization and Specialization associations with the class GF_InheritanceRelation. Instead, the class has an association with itself with the roles subType and superType. GF_InheritanceRelation is not realised in the S-100 GFM;

The multiplicity of the superType is 0..1 to represent the concept that a feature may have a maximum of one superType. This is in order to prevent multiple-inheritance in S-100;

The multiplicity of the role carrierOfCharacteristics with S100_GF_PropertyType (the S-100 realisation of GF_PropertyType) is changed from 0..* to 1..*. An S-100 feature must have properties.

Table 3-3— S100_GF_FeatureType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_FeatureType	A type for an abstract representation of a real world phenomenon.	-	-
Role	superType	The more generic feature type from which this feature type is derived.	0..1	S100_GF_FeatureType
Role	subType	The more specific feature types which are derived from this feature type.	0..*	S100_GF_FeatureType
Role	linkBetween	A link to a feature association that specify the relationship between one feature type and the same or another feature type.	0..*	S100_GF_FeatureAssociationType
Role	carrierOfCharacteristics	Attributes and roles that describe the characteristics of a feature type.	1..*	S100_GF_PropertyType

3-5.2.7 S100_GF_PropertyType

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

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

Table 3-4 — S100_GF_PropertyType

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

3-5.2.8 S100_GF_AttributeType

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

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

Table 3-5— S100_GF_AttributeType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_ AttributeType	Abstract base class for all attributes of feature types. In this model are two sub classes: thematic attributes and spatial attributes.	-	-
Attribute	valueType	The data type of the attribute value.	1	CharacterString
Attribute	domainOfValues	Description of a set of values. For codelist types this may be a URI identifying a "vocabulary."	1	CharacterString
Attribute	multiplicity	The number of instances of the attribute that may be associated with a single instance of a feature type	1	S100_Multiplicity

3-5.2.9 S100_GF_AssociationRole

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

Table 3-6 — S100_GF_AssociationRole

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

3-5.2.10 GF_Operation

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

3-5.2.11 S100_GF_AssociationType

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

1) The ISO 19109 GFM models GF_AssociationType as a subtype of the class GF_FeatureType. This is done for reasons which are set out in Note 1 of ISO 19109 clause 7.3.9. The S-100 model does not model the class as a subtype of S100_GF_FeatureType. Within S-100 associations between feature types are not considered abstractions of real world phenomena. The result of this approach to modelling the GFM is that the only properties associations can have are thematic attributes.
2) The multiplicity of roleName is 0..2 instead of 1..*. The lower bound of 0 means the role is one of the default roles “source” or “target” and this is obvious from the application schema’s semantics of the association type’s name and the names of the participating feature or information classes. The upper bound expresses the constraint that S100 does not allow associations with more than two participating classes.

Table 3-6— S100_GF_AssociationType

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

3-5.2.12 S100_GF_InformationType

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

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

Table 3-7 — S100_GF_InformationType

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

Role	roles	Roles for associations to other information type that supplying supplementary information.	0..*	S100_GF_AssociationRole

3-5.2.13 S100_GF_FeatureAssociationType

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

Table 3-8— S100_GF_FeatureAssociationType

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

3-5.2.14 S100_GF_InformationAssociationType

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

Table 3-9— S100_GF_InformationAssociationType

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

3-5.2.15 S100_GF_Constraint

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

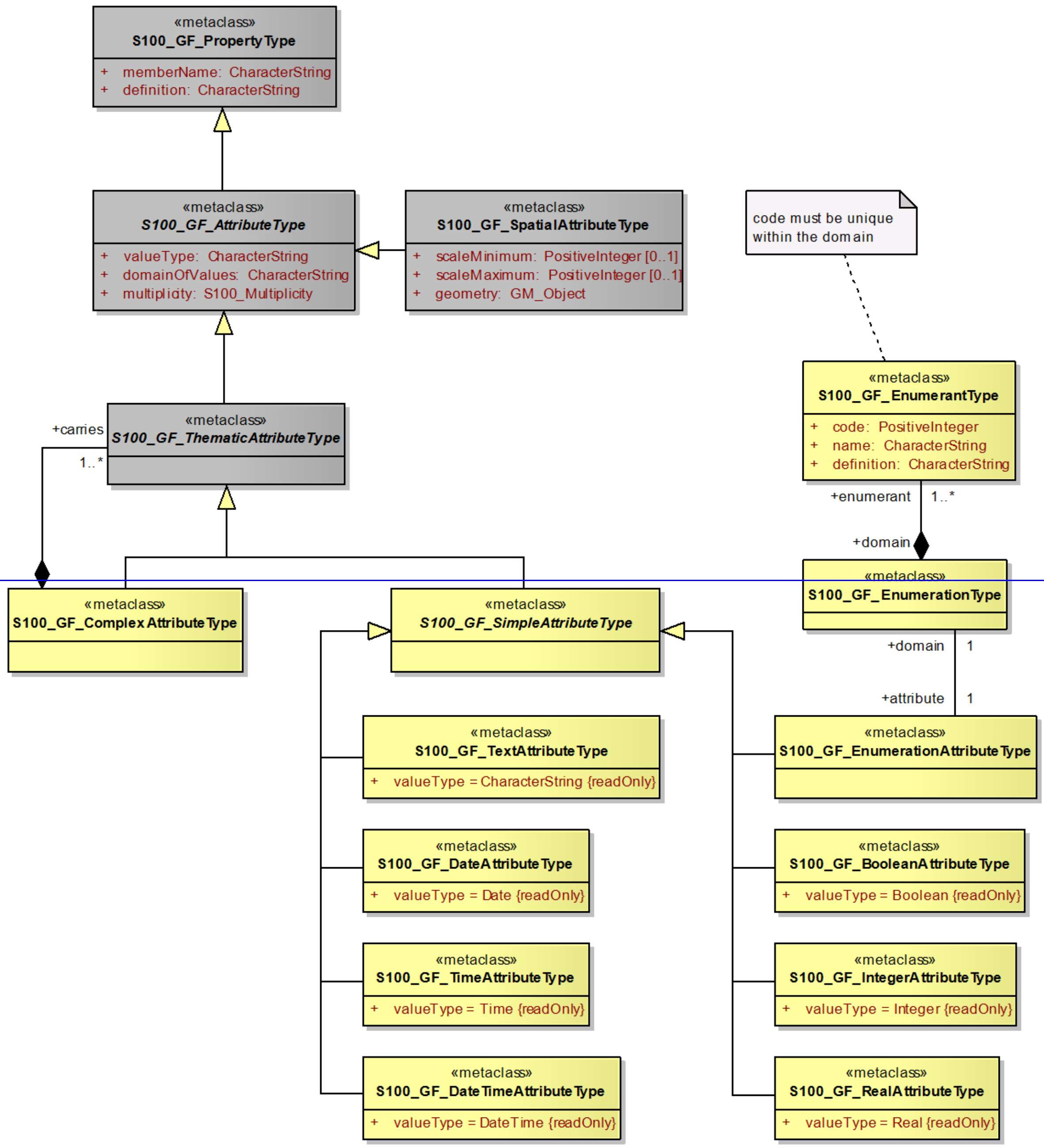
Table 3-10— S100_GF_Constraint

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

3-5.3 Attributes of feature types

3-5.3.1 Introduction

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


«metaclass»
S100_GF_PropertyType
+ memberName: CharacterString
+ definition: CharacterString
«metaclass»
S100_GF_AttributeType
+ valueType: CharacterString
+ domainOfValues: CharacterString
+ multiplicity: S100_Multiplicity
«metaclass»
S100_GF_SpatialAttributeType
+ scaleMinimum: PositiveInteger [0..1]
+ scaleMaximum: PositiveInteger [0..1]
+ geometry: GM_Object
code must be unique within the domain
«metaclass»
S100_GF_EnumerantType
+ code: PositiveInteger
+ name: CharacterString
+ definition: CharacterString
+enumerant
1..*
+domain
+carries
1..*
«metaclass»
S100_GF_ThematicAttributeType
«metaclass»
S100_GF_EnumerationType
«metaclass»
S100_GF_ComplexAttributeType
«metaclass»
S100_GF_SimpleAttributeType
+domain
1
+attribute
1
«metaclass»
S100_GF_TextAttributeType
+ valueType = CharacterString {readOnly}
«metaclass»
S100_GF_EnumerationAttributeType
«metaclass»
S100_GF_DateAttributeType
+ valueType = Date {readOnly}
«metaclass»
S100_GF_BooleanAttributeType
+ valueType = Boolean {readOnly}
«metaclass»
S100_GF_TimeAttributeType
+ valuetype = Time {readOnly}
«metaclass»
S100_GF_IntegerAttributeType
+ valueType = Integer {readOnly}
«metaclass»
S100_GF_DateTimeAttributeType
+ valueType = DateTime {readOnly}
«metaclass»
S100_GF_RealAttributeType
+ valueType = Real {readOnly}

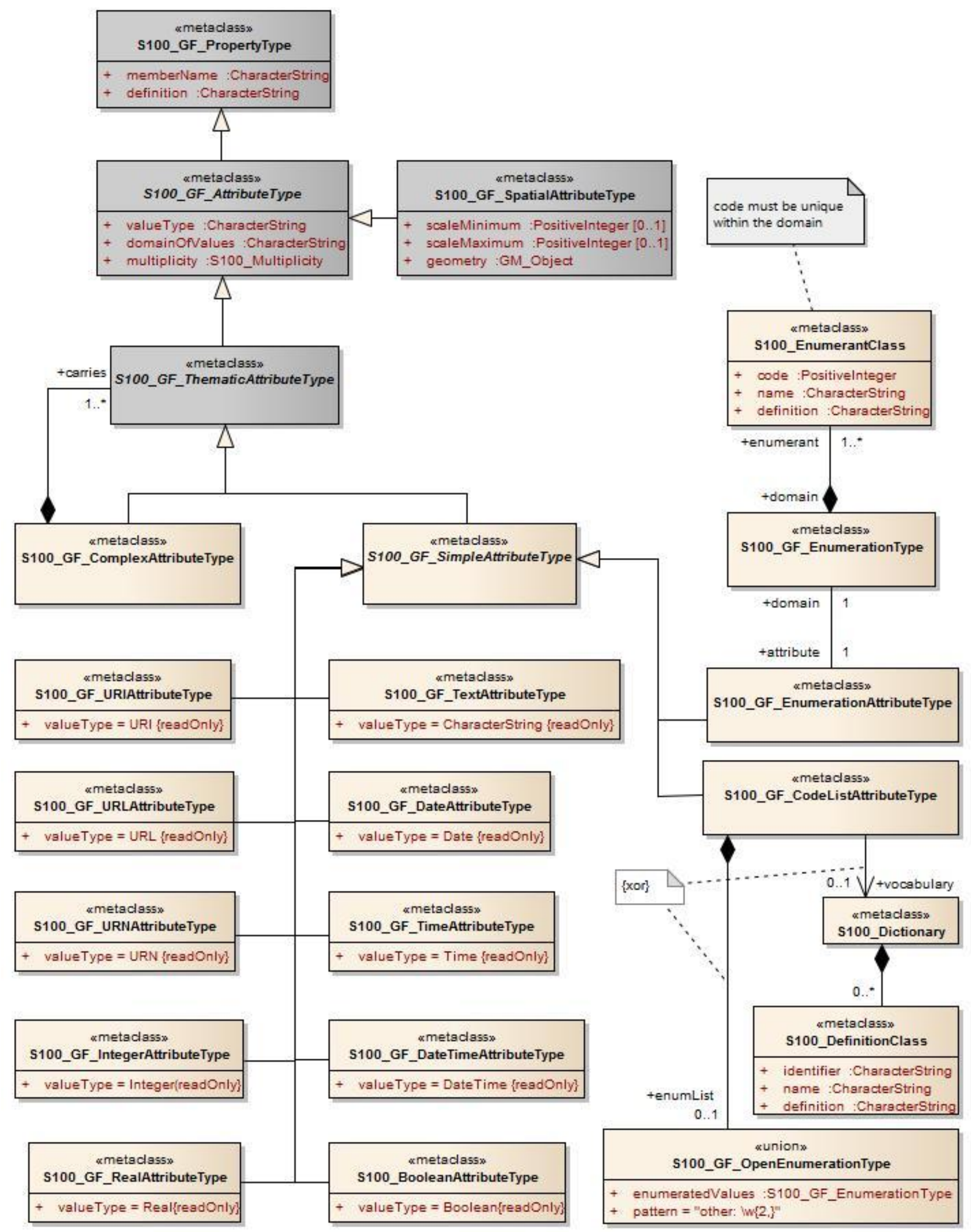


Figure3-2 — Attributes

3-5.3.2 S100_GF_ThematicAttributeType

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

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

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

Table 3-11— S100_GF_ThematicAttributeType

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

3-5.3.3 S100_GF_ComplexAttributeType

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

3-5.3.4 S100_GF_SimpleAttributeType

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

3-5.3.5 S100_GF_SpatialAttributeType

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

Table 3-12— S100_GF_SpatialAttributeType

Role Name	Name	Description	Mult.	Type
Class	S100_GF_ SpatialAttributeType	Class representing a spatial attribute, which shall be used to express spatial characteristics of a feature type.	-	-
Attribute	scaleMinimum	The smallest denominator of a scale for that an instance of a feature type shall be used (e.g. for portrayal)	0..1	PositiveInteger
Attribute	scaleMaximum	The largest denominator of a scale for that an instance of a feature type shall be used (e.g. for portrayal)	0..1	PositiveInteger
Attribute	geometry	The object that describes the geometry of an instance of a feature type.	1	GM_Object

3-5.3.6 GF_TemporalAttributeType

The ISO 19109 class GF_TemporalAttributeType is not realised explicitly in the S-100 GFM. Temporal information shall be modelled using the thematic attribute type S100_GF_ThematicAttributeType (see section 6.3.3 for more details).

3-5.3.7 GF_MetadataAttributeType

The ISO 19109 class GF_MetadataAttributeType is not realised explicitly in the S-100 GFM. Metadata types shall be modelled using complex thematic attributes which realise types from the S-100 Part 3 metadata component. The complex thematic attributes shall be defined in a feature catalogue.

3-5.3.8 GF_QualityAttributeType

The ISO 19109 class GF_QualityAttributeType is not realised explicitly in the S-100 GFM. Quality metadata types shall be modelled using complex thematic attributes which realise types from the S-100 Part 4C Appendix 4C-A Data Quality. The complex thematic attributes shall be defined in a feature catalogue.

3-5.3.9 GF_LocationAttributeType

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

3-5.4 Relationships between named types

3-5.4.1 Introduction

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

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

3-5.4.2 GF_InheritanceRelation

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

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

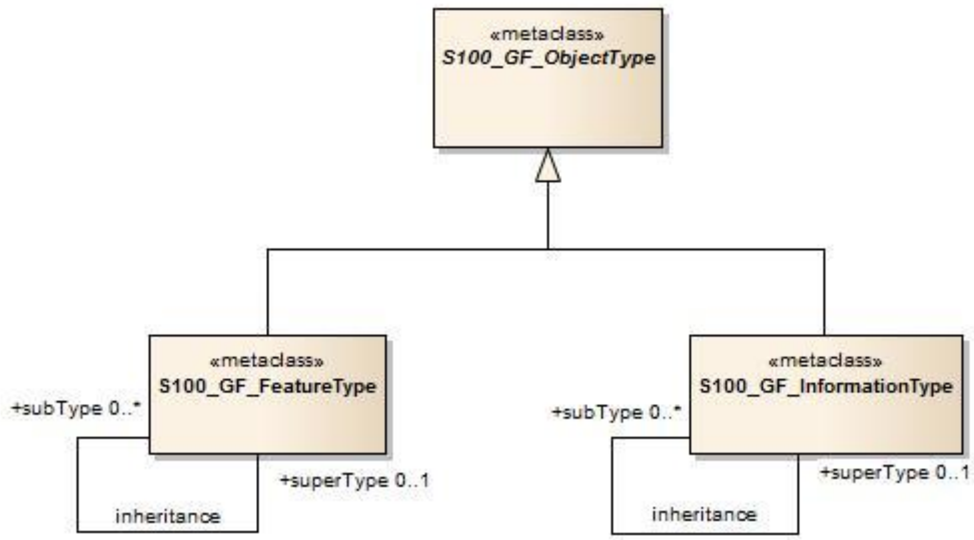


Figure 3-3 — Specialisation and Generalisation Associations

3-5.4.3 S100_GF_AssociationType

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

3-5.4.4 Associations to information types

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

3-5.4.5 Default names for association ends

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

1) If only one end of an association is given an explicit name “<rolename>”, the other end shall have the default name “inv_<rolename>”.

2) If neither end of the association is given an explicit name, the default role name is "the<target class name>" in which the target class is referenced from the source class.
3) The above rules result in a distinct name for each association end in an application schema. If standard names are desired, the following defaults may be used instead of those listed above.
    a. The role "additionalInformation" is a default role name for associations from feature to information types.
    b. Feature/feature or information/information associations navigable in only one direction may use the default end names "source" and "target". The name "associatedWith" may be used at both ends of a bidirectional association.

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

3-5.5 Behaviour of feature types

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

3-5.6 Constraints

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

Only named types and properties may have constraints.

3-6 Rules for application schema (ISO 19109 Clause 8)

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

The application schema serves two purposes:

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

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

Surveying the requirements from the intended field of application (Universe of Discourse).

1) Making a conceptual model of the application with concepts defined in the GFM. This task consists of identifying feature types, their properties and constraints.
2) Describing elements of the application schema in a formal modelling language where necessary. S-100 application schemas shall be described using the UML according to rules defined in this part of S-100.
3) Integrating the formal application schema with other standardized schemas, (spatial schema, quality schema, etc.) into a complete application schema.

3-6.2 The application schema (ISO 19109 Clause 8.2)

3-6.2.1 Conceptual schema language for application schemas

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

3-6.2.2 Main rules

The data structures of the application schema shall be modelled in the application schema.

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

3-6.2.3 Identification of application schemas

1) The identification of each application schema shall include a name and a version. The inclusion of a version ensures that a supplier and a user agree on which version of the application schema describes the contents of a particular dataset. A system of defining unique names and versions for S-100 application schemas shall be defined.
2) In UML, an application schema shall be described within a PACKAGE, which shall carry the name of the application schema and the version stated in the documentation of the PACKAGE.

3-6.2.4 Documentation of an application schema

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

3-6.3 Rules for application schema in UML (ISO 19109 Clause 8.3)

3-6.3.1 Main rules (ISO 19109 Clause 8.3.1)

The main rules for application schemas in UML are:

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

7) An instance of S100_GF_AttributeType shall be implemented as an ATTRIBUTE.
8) An instance of S100_GF_SimpleAttributeType shall be implemented as an ATTRIBUTE.
9) An instance of S100_GF_ComplexAttributeType shall be implemented as a CLASS. The instantiated CLASS shall have one or more instances of S100_GF_SimpleAttributeType as its ATTRIBUTE(s) and or S-100_GF_ComplexAttributeType
10) An instance of the association inheritanceRelation shall be represented by a UML GENERALISATION relationship.

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

3-6.4.1 Rules for adding information to a standard schema

Standard schemas shall not be extended within application schemas. Standard schemas are those that are documented in S-100 e.g the spatial schema, feature catalogue schema etc.

3-6.4.2 Restricted use of standard schemas

For some standard schemas, e.g. S-100 Part 8 (spatial schema), it is possible to redefine the schema in such a way that only selected parts of the schema will be used, and only some of the definitions of classes and relationships will be used.

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

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

The metadata schema defined in S100 Part 5 is an application schema for metadata data sets. Metadata are data describing and documenting data. Metadata for geographic data typically provides information about their identification, extent, quality, spatial and temporal aspects, spatial reference and distribution.

Metadata types shall be implemented as complex attributes that realise elements from S100 Part 5. Thus metadata attributes shall be thematic attribute types.

3-6.4.4 Temporal rules (ISO 19109 Clause 8.6)

S-100 does not includeuses a profile of ISO 19108 described in clause 3-8. Temporal attributes shall be modelled using the types Date, Time or DateTime, S100_TruncatedDateTime, or complex attributes using combinations of these temporal types or the complex type S100_IndeterminateDateTime. Use of these types makes the attribute an instance of S100_GF_SimpleAttributeType or S100_GF_ComplexAttributeType, as appropriate.

3-6.5 Spatial rules (ISO 19109 Clause 8.7)

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

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

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

3-6.5.2 Spatial attributes

1) Spatial characteristics of a feature shall be described by one or more spatial attributes. In an application schema, a spatial attribute is a subtype of a feature attribute (see 5.3), and the taxonomy of its values is defined in the S-100 Part 8.
2) A spatial attribute shall be represented in an application schema in either of two ways:

a) Case 1: as an ATTRIBUTE of a UML CLASS that represents a feature, in which case the ATTRIBUTE shall take one of the spatial objects defined in the spatial schema, ISO 19107, as the data type for its value, or

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

3) spatial attribute shall take a spatial object as its value. Spatial objects are classified as geometric objects, which are sub-classed as primitives, complexes or aggregates (for geometric objects). Table 1 lists spatial objects that shall be used in an application schema as values for spatial attributes.

3-6.5.3 Spatial Quality

The positional quality of a spatial object shall be described by a one way association to a S100_GF_InformationType which is associated with a S100_GF_SimpleAttribute carrying positional accuracy.

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

3-6.5.4.1 Introduction

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

3-6.5.4.2 Geometric aggregates

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

3-6.5.4.3 Geometric complexes

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

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

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

3-6.5.4.4 Geometric composites

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

3-6.5.4.5 Features sharing geometry

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

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

1) An application schema may require instances of two or more feature types to share their geometry completely by including a constraint that the GM_Objects representing the features must be equal.
2) An application schema may preclude instances of two or more feature types from sharing their geometry completely by including a constraint that the GM_Objects representing the features are not equal.

3-6.6 Cataloguing rules (ISO 19109 Clause 8.8)

3-6.6.1 Introduction (ISO 19109 Clause 8.8.1)

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

3-6.6.2 Application schema based on a feature catalogue (ISO 19109 Clause 8.8.2)

An S-100 application schema shall be completely constructed by the definitions provided by a feature catalogue implementing the S-100 feature catalogue profile.

3-6.6.3 Character encoding

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

3-7 Application Schema for Coverages

3-7.1 Introduction

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

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

3-7.2 Gridded Data

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

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

This template application schema supports the majority of imagery and gridded data applications.

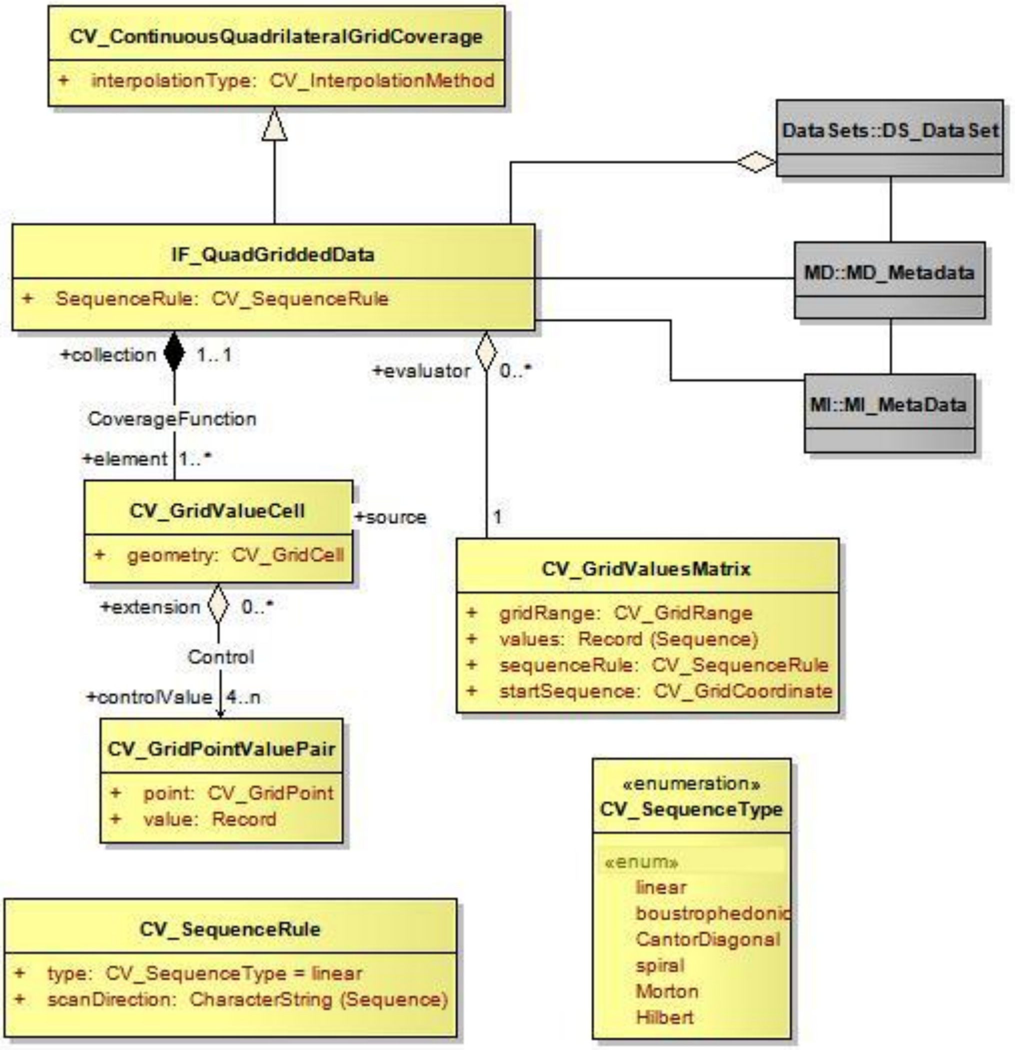


Figure 3-4 – Template application schema for a Quadrilateral Grid Coverage

3-7.3 Variable Cell Size Grid

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

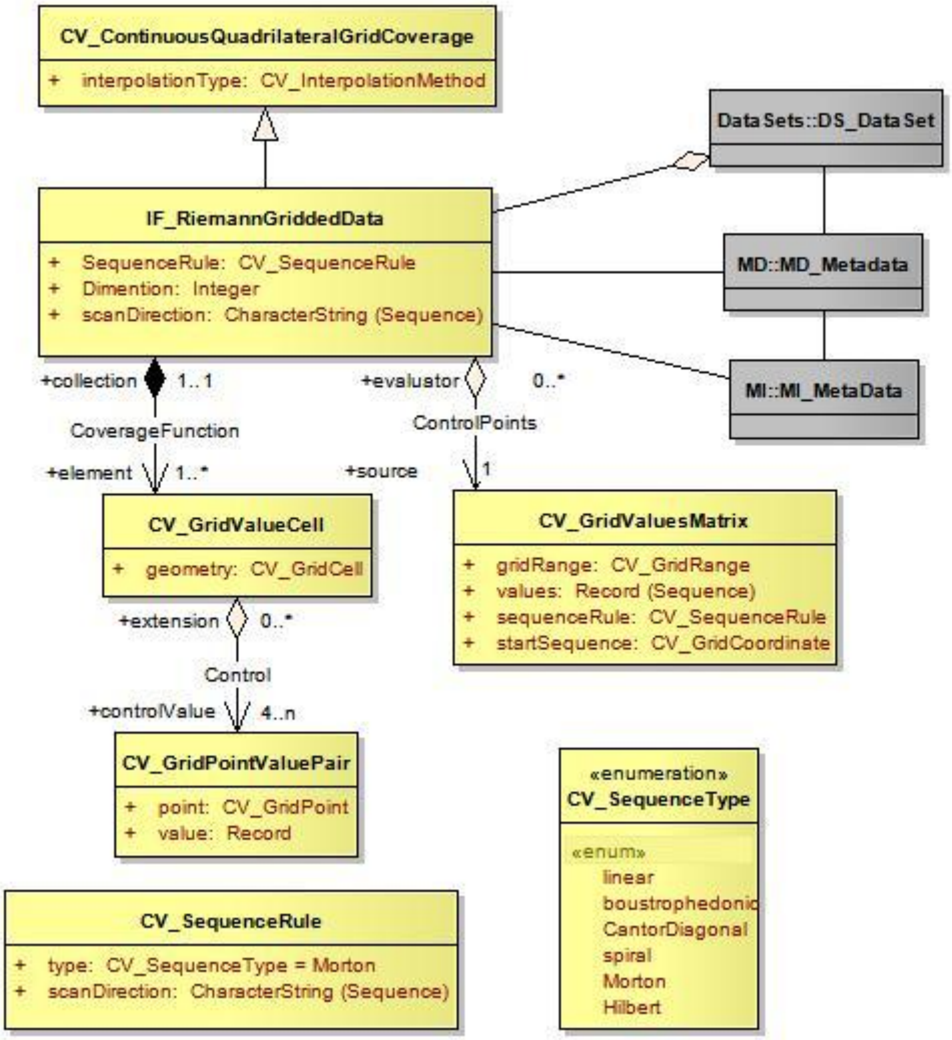


Figure 3-5 – Template application schema for a Reimann Grid Coverage

3-7.4 Feature Oriented Image

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

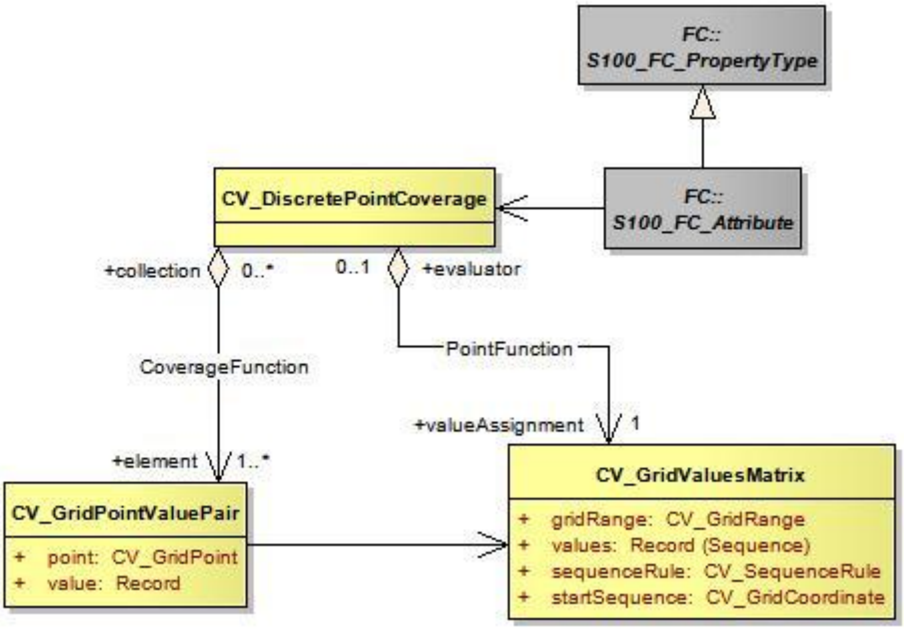


Figure 3-6 – Feature oriented discrete coverage

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

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

3-8 S-100 Temporal Information

3-8.1 Introduction

ISO 19108 provides the concepts needed to describe the temporal characteristics of geographic information as they are abstracted from the real world. Temporal characteristics of geographic information include attributes, associations and metadata elements that take a value in the temporal domain. Time provides a fundamental element within many geographic datasets. Consistent modelling of temporal information is required to ensure consistent interaction between different S-100 products and across domains.

3-8.2 Temporal Schema

The temporal schema consists of temporal objects and temporal reference system. Temporal objects defines temporal geometric and topological objects that shall be used as values for the temporal characteristics of features and data sets. The temporal position of an object shall be specified in relation to a temporal reference system. S-100 products shall use the Gregorian Calendar and 24-hour local or Coordinated Universal Time (UTC) for information interchange as specified in ISO 8601. Where local time is used the offset from UTC must be provided see Value Types.

3-8.3 TM Objects

S-100 constrains temporal objects to a subset of those defined by ISO 19108. TM_Object (see Figure 1) is an abstract class that has two subclasses defined in ISO 19108, of which S-100 uses only TM_Primitive. TM_Primitive is an abstract class that represents a non-decomposed element of geometry or topology of time. TM_GeometricPrimitive provides information about temporal position.

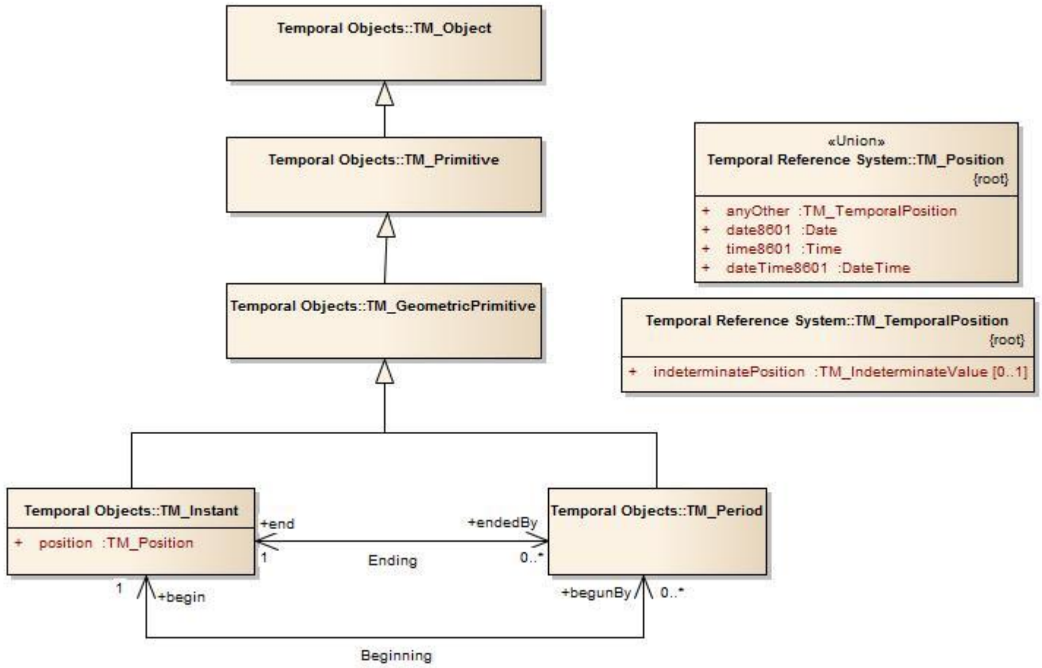


Figure 3-7 – Temporal Objects model extracted from ISO 19108

3-8.4 Temporal Geometric Primitives

The two geometric primitives in the temporal dimension are the instant and the period. These primitives are defined analytically in the case of time measured on an interval scale, and analogically in the case of time measured on an ordinal scale. TM_GeometricPrimitive is an abstract class with two subclasses, TM_Instant represents an instant and TM_Period represents a period (see Figure 1).

3-8.5 S100_TM_Instant

An instant is a zero-dimensional geometric primitive that represents position in time. It is equivalent to a point in space. In practice, an instant is an interval whose duration is less than the resolution of the time scale.

Attributes:

TM_Instant has one attribute

position:TM_TemporalPosition shall provide the position of this TM_Instant. An instance of TM_Instant is an identifiable object, while an instance of TM_TemporalPosition is a data value.

3-8.6 S100_TM_Period

The period is a one-dimensional geometric primitive that represents extent in time. The period is equivalent to a curve in space. Like a curve, it is an open interval bounded by beginning and end points (instants), and has length (duration). Its location in time is described by the temporal positions of the instants at which it begins and ends; its duration equals the temporal distance between those two temporal positions. Since it is impossible to measure duration on an ordinal scale, an instant cannot be distinguished from a period on this basis. In practice, the time at which a single event occurs can be considered an instant when time is measured.

a) *position:TM_TemporalPosition* shall provide the position of this TM_Instant. An instance of TM_Instant is an identifiable object, while an instance of TM_TemporalPosition is a data value. A series of consecutive events must occupy an interval of time, which is a period. The term period is commonly applied to sequences of events that have distinctive characteristics in common.

Associations:

a) *Beginning* links the TM_Period to the TM_Instant at which it starts.

b) *Ending* links the TM_Period to the TM_Instant at which it ends.

Constraints:

a) self.begin.position < self.end.position states that the temporal position of the beginning of the period must be less than (i.e. earlier than) the temporal position of the end of the period.

3-8.7 S100_TM_Position

TM_Position as defined in ISO 19108 is a union class that consists of one of the data types listed as its attributes. Date, Time, and DateTime are basic data types defined in ISO/TS 19103. S100_TM_TemporalPosition adds a type for truncated representations. Date, Time, and DateTime comply with ISO 8601 encoding of dates and times as character strings. These data types may be used for describing temporal positions referenced to the Gregorian calendar and UTC. The data types defined in 5.4.4 specify numeric values for dates and times. They may be used for temporal positions referenced to any calendar or clock, including the Gregorian calendar and UTC.

3-8.8 S100_TM_TruncatedDateTimeType

S-100 extends ISO 19108 to include a specific data type for truncated date time. This ensures that partial dates can be used for recurring periods. The format of this type is as follows:

YYYYMMDDT*HHMMSS*

YYYY Year

MM Month

DD Day

T Time designator (character ‘T’)

HH Hours

MM Minutes

SS Seconds

At least one value must be present and truncated values represented with a hyphen. The time component is optional and the time designator is included if and only if a time component is present.

3-8.9 S100_TM_IndeterminateDateTime

Indeterminate instants are encoded using the complex type S100_TM_IndeterminateDateTime, derived from S100_TM_TruncatedDateTimeType. This derived type designates an instant related to a given date-time value by one of the temporal relations ‘before’ or ‘after’. The temporal relation is given by the enumerated type attribute *indeterminatePosition*.

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

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

3-8.10 S-100 Temporal Model

Figure 3-8 describes the temporal information model.

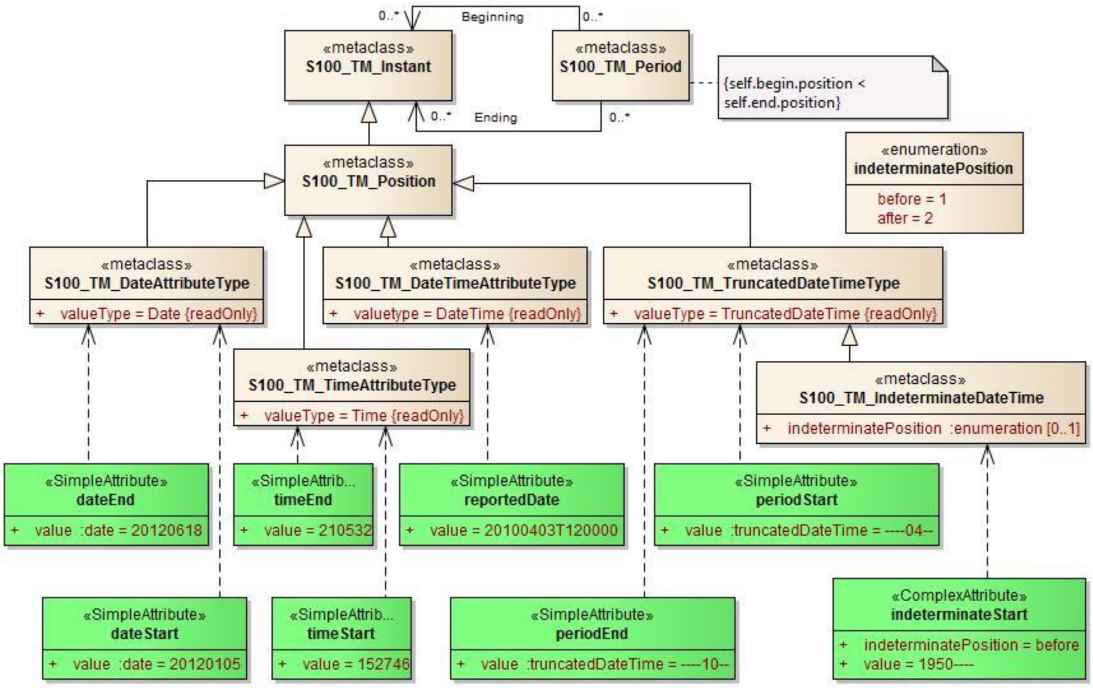


Figure 3-8 - S-100 Temporal Information Model – Items in green are indicative examples only.

3-8.11 Value types

Date, Time, DateTime, and S100_TruncatedDateTime are basic types described in the table below.

Table 1. Value types

Date	A date gives values for year, month and day according to the Gregorian Calendar. Character encoding of a date is a string which shall follow the calendar date format (complete representation, basic format) for date specified by ISO 8601. EXAMPLE 19980918 (YYYYMMDD)
Time	A time is given by an hour, minute and second or fractions thereof. Character encoding of a time is a string that follows any of the time of day formats defined in ISO 8601. Product specifications shall specify which formats are allowed for their domains, and where appropriate, the decimal separator and number of digits in the decimal fraction. Time zone according to UTC is optional. EXAMPLES: a) 183059 or 183059+0100 or 183059Z (complete representation,

	basic format) b) 18:30:59 or 18:30:59+0100 or 18:30:59Z (complete representation, extended format) c) 1830 or 1830+0100 or 1830Z (reduced accuracy with 2 digits omitted, basic format) d) 18:30 or 18:30+0100 or 18:30Z (reduced accuracy with 2 digits omitted,, extended format) e) 18 or 18Z (reduced accuracy with 4 digits omitted, basic format – extended format is not applicable when 4 digits are omitted). f) 183059.50, 1830.7, 18.50 (decimal representations, basic format) g) 18:30:59.50, 18:30.7 (decimal representations, extended format) Note that the time designated by (c) and (d) is different from the time designated by (a) and (b) and the time designated by (e) is different from both. The complete representation of the time of 27 minutes and 46 seconds past 15 hours locally in Geneva (in winter one hour ahead of UTC), and in New York (in winter five hours behind UTC), together with the indication of the difference between the time scale of local time and UTC, are used as examples. Geneva: 152746+0100 New York: 152746-0500
DateTime	A DateTime is a combination of a date and a time type. Character encoding of a DateTime shall follow ISO 8601 (see above). EXAMPLE: 19850412T101530
S100_TM_TruncatedDateTime	A TruncatedDateTime allows a partial TM Position to be given. At least one of the following components must be present with omitted elements replaced by the appropriate number of hyphens. year – integer between 0000 - 9999 month – integer between 01-12 day – integer between 01 and 28, 29, 30, or 31 depending on year and month values time – Time type (see above)

S100_IndeterminateDateTime is a complex attribute type with two sub-attributes:

Sub-attribute	label	Type	Remark
indeterminatePosition	Indeterminate time position	enumeration	1: before 2: after
value	Referenced time	S100_TruncatedDateTime	

3-8.12 Interpretation of interval start and end

The start and end instants of periods shall be included in the period unless a product specification specifies a different interpretation. This is based on ISO 8601:2004 § 2.1.3 which defines time interval as "the part of the time axis delimited by two instants" and provides that "A time interval comprises all instants between the two limiting instants and, unless otherwise stated, the two limiting instants themselves".

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

EXAMPLES: Applying this to encoding intervals using the reduced accuracy representation or the truncatedDateTimeType, results in the interpretations in Table 2. The table also indicates how the special case of leap years can be handled.

Table 2. Examples of periods

<S100_TM_TruncatedDateTimeType> periodStart	`----01--` year, day, and time not encoded	000000 on January 1 through 240000 on the 29th day of February in leap years and the 28th day of February in non-leap years
<S100_TM_TruncatedDateTimeType> periodEnd	`----02--` year, day, and time not encoded	
<S100_TM_TruncatedDateTimeType> periodStart	`----0101` year and time not encoded	000000 on January 1 through 240000 on the 28th day of February each year
<S100_TM_TruncatedDateTimeType> periodEnd	`----0228` year and time not encoded	
<S100_TM_DateAttributeType> dateStart	dateStart=20120105	000000 on January 5, 2012 through 240000 on June 18, 2012
<S100_TM_DateAttributeType> dateEnd	dateEnd=20120618	

3-8.13 Use of specific types for truncated Date Time

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

Example;

An XML based encoding may use the *gMonthDay* simple attribute type (which is an XML Schema built-in type).

xs:gMonthDay: `--12-17`

The S-100 truncatedDateTime value is `----1217`.