ISO TC 211/SC N XXXX

Date: 2006-07-28

ISO/WD

ISO TC 211/SC /WG

Secretariat: SNV

DIPWG1-14.1A

Joint 18th TSMAD & 1st DIPWG Meeting
Ottawa, Canada, 4-8 May 2009

Geographic information — Portrayal

Information géographique — Présentation

Warning

This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Document type: International Standard
Document subtype:
Document stage: (20) Preparatory
Document language: E

H:\pac\DIPWG\DIPWG1\Documents\DIPWG1-14.1A_ISO_19117r1-E-2008-12-25.doc STD Version 2.1c2

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

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

ISO was prepared by Technical Committee ISO/TC 211, *Geographic information/Geomatics*, Subcommittee SC , .

This second/third/... edition cancels and replaces the first/second/... edition (), [clause(s) / subclause(s) / table(s) / figure(s) / annex(es)] of which [has / have] been technically revised.

Introduction

This International Standard specifies a conceptual schema for portrayal data, in particular symbols and portrayal rule sets. Portrayal rule sets associate features with symbols for the portrayal of the features on maps and other display media. This schema includes classes, attributes, associations and operations that provide a common conceptual framework that specifies the structure of and interrelationships between features, portrayal rule sets, and symbols. This framework specifies the concepts found in existing portrayal implementations and standardizes these concepts for use in future implementations.

Geographic information — Portrayal

1 Scope

This International Standard specifies a conceptual schema for describing symbols, collections of symbols, and portrayal rule sets, which are collections of rules that map features to symbols in a generic, dataset independent manner.

This International Standard does not define a standard symbol collection, or a standard for symbol graphics. It also does not address any of the services that actually portray feature data.

2 Conformance

TBD

3 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 19101:2002 *Geographic information — Reference model*

ISO/TS 19103:2005 *Geographic information — Conceptual schema language*

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 19111:2003, *Geographic information — Spatial referencing by coordinates*

ISO 19115:2003, *Geographic information — Metadata*

ISO 19123:2005 *Geographic information — Schema for coverage geometry and functions*

ISO 19133:2005 *Geographic information — Location-based services — Tracking and navigation*

ISO/IEC 19501-1:2005 *Information technology — Unified Modelling Language (UML) — Part 1: Specification*

4 Terms, definitions and abbreviated terms

4.1 Terms and definitions

For the purposes of this International Standard, the following terms and definitions apply.

4.1.1
annotation
any marking on illustrative material for the purpose of clarification, such as numbers, letters, symbols, and signs

4.1.2
catalogue
a complete organized list of items in a collection facilitating the location and retrieval of the items

4.1.3
class
description of a set of objects that share the same attributes, operations, methods, relationships, and semantics

[ISO/TS 19103]

NOTE A class may use a set of interfaces to specify collections of operations it provides to its environment. See: interface.

4.1.4
curve
1-dimensional geometric primitive, representing the continuous image of a line

[ISO 19107]

NOTE The boundary of a curve is the set of points at either end of the curve. If the curve is a cycle, the two ends are identical, and the curve (if topologically closed) is considered to not have a boundary. The first point is called the start point, and the last is the end point. Connectivity of the curve is guaranteed by the "continuous image of a line" clause. A topological theorem states that a continuous image of a connected set is connected.

4.1.5
dataset
identifiable collection of data

[ISO 19115]

NOTE The principles which apply to datasets may also be applied to dataset series and reporting groups.

4.1.6
delineation
the geometric dimension associated with the spatial attribute that is used to portray a feature

4.1.7
external function
function not part of the application schema

NOTE The electronic map in a car navigation system has to be displayed so that the up-direction of the map is always in the direction the car is moving. To be able to specify the rotation of the map, the current position of the car must be retrieved continuously from an external position device using an external function.

4.1.8
feature
abstraction of real world phenomena

[ISO 19101]

NOTE A feature may occur as a type or an instance. Feature type or feature instance should be used when only one is meant.

4.1.9
feature attribute
characteristic of a feature

[ISO 19101]

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’.

NOTE 1 A feature attribute has a name, a data type, and a value domain associated to it. A feature attribute for a feature instance also has an attribute value taken from the value domain.

NOTE 2 In a feature catalogue, a feature attribute may include a value domain but does not specify attribute values for feature instances.

4.1.10
feature portrayal rule set
collection of portrayal rules that apply to a feature instance

4.1.11
geographic information
information concerning phenomena implicitly or explicitly associated with a location relative to the Earth

[ISO 19101]

4.1.12
geometric primitive
geometric object representing a single, connected, homogeneous element of space

[ISO 19107]

4.1.13
instance
object that realizes a class

[ISO 19107]

4.1.14
interface
named set of attributes and operations that characterize the behaviour of an element

[adapted from ISO/TS 19103]

4.1.15
library
a collection of a single type of item accessible for use

4.1.16
mapping
a rule that assigns to each element in a set a specific element in a second set

4.1.17
metadata
data about data

[ISO 19115]

4.1.18
point
0-dimensional geometric primitive, representing a position

[ISO 19107]

4.1.19
portrayal
presentation of information to humans

4.1.20
portrayal mapping
a mapping from features to symbols

4.1.21
portrayal rule
rule that is applied to the feature to determine what symbol to use

4.1.22
spatial attribute
feature attribute describing the spatial representation of the feature by coordinates, mathematical functions and/or boundary topology relationships

4.1.23
surface
2-dimensional geometric primitive, locally representing a continuous image of a region of a plane

[ISO 19107]

4.1.24
symbol
a graphic element of a portrayal that represents an instance of a feature and may also reflect elements of a feature's attribution

4.2 Abbreviated terms

BNFBackus Naur Form

OCL Object Constraint Language (ISO 19501)

UML Unified Modelling Language (ISO 19501)

5 UML notation

TBD

6 Enterprise view of portrayal

Figure 1 — Enterprise View of Portrayal

Table 1 — Portrayal Policies

7 Portrayal principles

7.1 Introduction

This International Standard defines a feature-centred rule based portrayal mechanism. Instances of features are portrayed based on rules, which make use of geometry and attribute information. The relationship between the feature instances, attributes and the underlying spatial geometry is specified in an application schema according to ISO 19109. Spatial geometry and associated topological relationships are defined in ISO 19107.

Portrayal specifications and portrayal rules are needed to portray a dataset containing geographic data. Portrayal specifications are selected according to specific portrayal rules (see clause **Erreur ! Source du renvoi introuvable.**). The portrayal mechanism makes it possible to portray the same dataset in different ways without altering the dataset itself. The portrayal mechanism is illustrated by Figure 2.

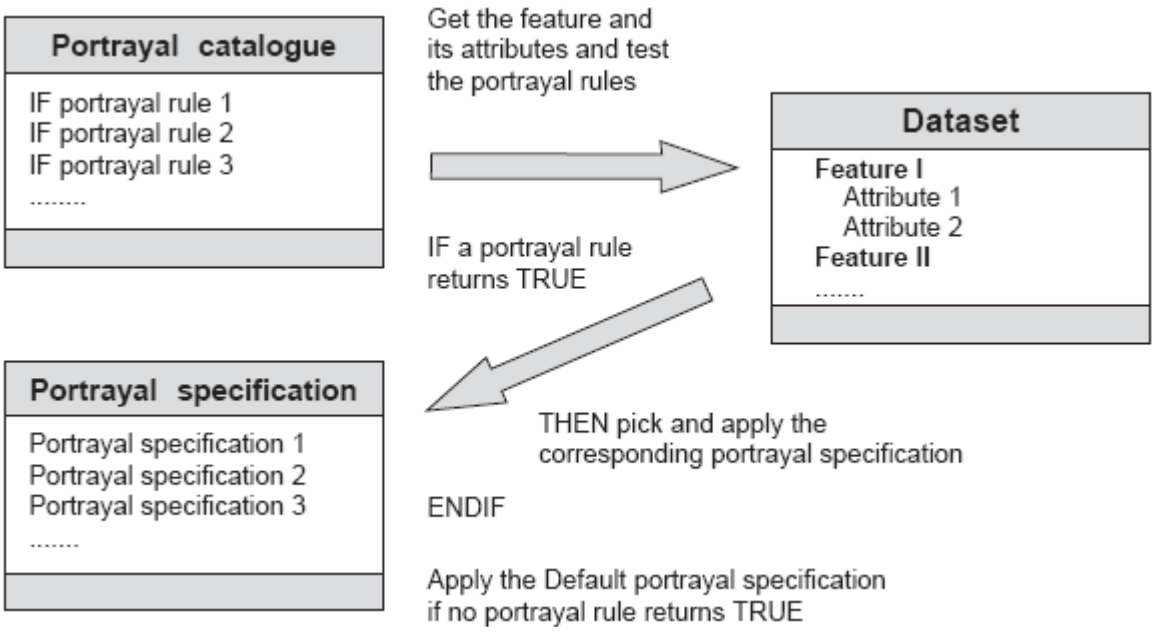

Figure 2 — The portrayal mechanism without priority attributes

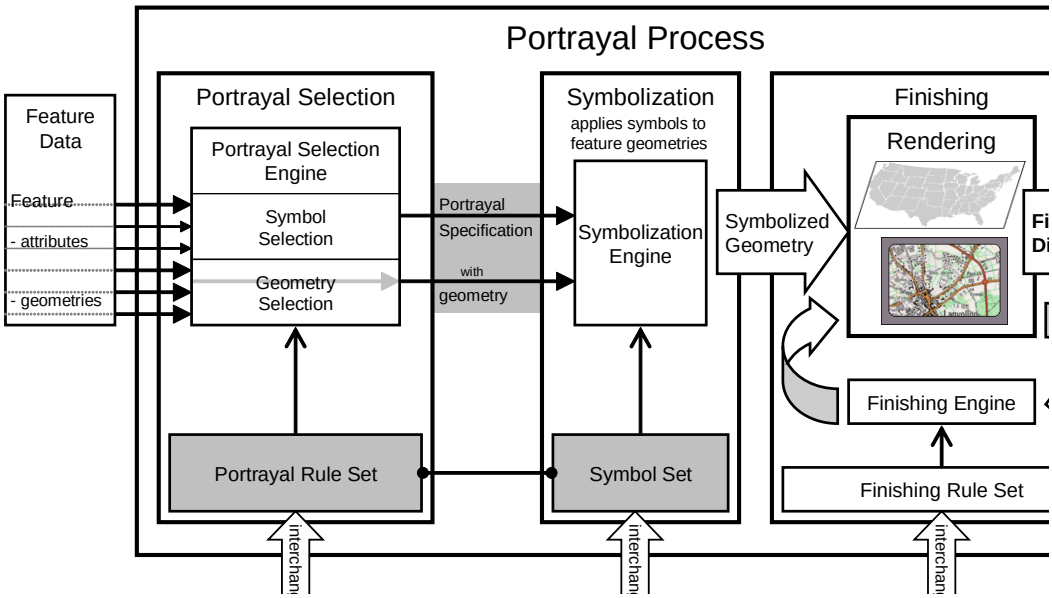

Figure 3 — Portrayal Process

Figure 4 — Portrayal structure Overview of portrayal

Figure 5 — UML of portrayal part of 19115

The portrayal specifications and portrayal rules shall not be part of the dataset. The portrayal rules shall be stored in a portrayal rule set. The portrayal specifications shall be stored separately from the dataset and referenced from the portrayal rules. The portrayal rules shall be specified for the feature class they will be applied on. The symbol definitions may be stored externally and referenced using a universal reference standard such as a network based URL.

Portrayal information may be specified either by sending a portrayal catalogue with the dataset, or by referencing an existing portrayal catalogue from Metadata. The model in Figure 3 shows how the portrayal catalogue is referenced by the dataset metadata. Only the metadata reference is shown and not the contents of the portrayal catalogue (see ISO 19115).

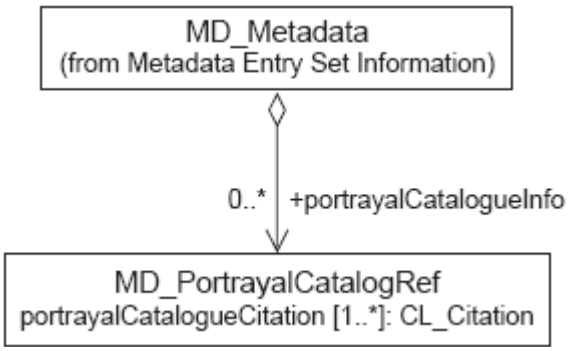

Figure 6 — UML model of the portrayal part of ISO 19115

In addition, the user may want to apply a user defined portrayal rule set and symbol definitions. This user defined portrayal rule set may also reference existing symbol definitions.

This standard does not mandate a specific rule expression language but a profile may specify a particular language.

The portrayal rules shall test the feature type, attributes, and delineation. of the feature instances in the dataset. An implementation may combine separate conceptual mappings into one rule set requiring that the resulting portrayal rules shall test the context values in addition to the feature type and attributes. The portrayal rule shall be applied as a query statement that returns TRUE or FALSE. The portrayal specification associated with the portrayal rule that returns TRUE shall be applied. If no portrayal rule returns TRUE then the default portrayal specification shall be used.

7.2 Portrayal schema mapping

7.2.1 Context

TBD

7.2.2 Rule based

Figure 7 — Rule Based Portrayal Mechanisms

7.2.3 Transformation

TBD

7.2.4 Population

TBD

7.2.5 Expressions

TBD

7.2.6 Feature mapping function

TBD

7.2.7 Portray nothing

TBD

7.2.8 Default portrayal specification

TBD

7.3 Legend (Portrayal Schema to Symbol Spec.)

TBD

7.4 Parameterization

TBD

7.5 Symbols

TBD

7.5.1 Composition

TBD

7.5.2 Parameterization of Symbols

TBD

7.6 Graphics

TBD

7.7 Annotation (non-feature portrayal)

TBD

7.8 Portrayal catalogues

TBD

7.9 Portray nothing

For a feature instance that is not to be portrayed, a portrayal rule shall return TRUE with associated portrayal specification that is empty when tested on the attributes of the feature instance (see 8.3.4). If no portrayal rule returns TRUE then the default portrayal specification shall be applied.

7.10 Default portrayal specification

The default portrayal specification shall be applied according to at least one of the spatial attributes of the feature instance, and shall only be applied when no portrayal rule returns TRUE' for a feature instance. A default portrayal specification shall be present for each delineation in a data set to ensure that no feature instance is left unportrayed by mistake.

7.11 Annotation

Typically there are two types of information: geographic information and annotation. Annotation includes text, grids, legends and special features such as a compass rose.

7.12 Overview of portrayal

Portrayal is illustrated by Figure 4. The diagram is not part of the portrayal schema and not for implementation. It is intended as an explanatory aid only.

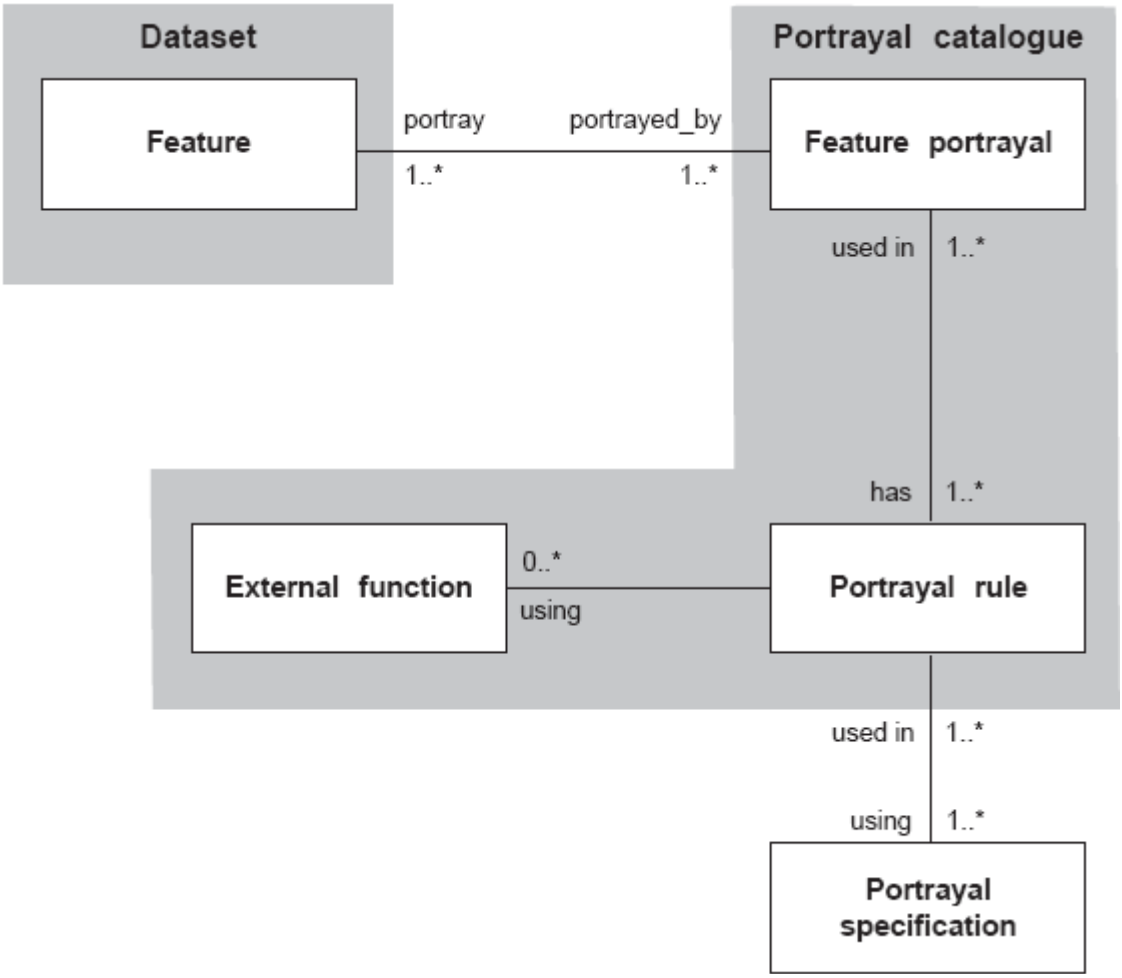

Figure 8 — Overview of portrayal

The portrayal catalogue consists of the feature portrayal, portrayal rule and external function, as shown in Figure 4.

To produce different portrayals of a data set, several different portrayal rule sets may be used. The same portrayal rule set may be used to portray one or more datasets. Dataset is explained in ISO 19109. The portrayal rule set relates to one or more symbol definitions, and one symbol definition may be referenced from one or more portrayal rule sets. A portrayal rule consists of two parts, a query statement, and an action statement, which may be simple or compound.

EXAMPLE 1 A Dataset contains instances of the feature class Road. The feature class Road contains two attributes, classification and segment. The classification attribute is of string data type, and may have the value “country road” or “town road”. The segment attribute is of GM_Curve type and contains the spatial description of the road. The Portrayal Specification used is called N50_specification. The two Portrayal Rules in this example look like this (The “quotes” in this example are used to show the contents of a string):

IF (Road.classification EQ “country road”) THEN drawCurve (“N50_specification.Solid_red_line”, Road.segment)

IF (Road.classification EQ “town road”) THEN drawCurve (“N50_specification.Solid_yellow_line”, Road.segment)

In this example the THEN separates the query and action statements. The drawCurve is an action statement drawing an actual curve using geometry from Road.segment and colour, line width etc. information from N50_specification.Solid_red_line and N50_specification.Solid_yellow_line.

7.13 Context

EXAMPLE 2 If the portrayal varies with the scale, an External Function is needed as part of the query statement. One of the Portrayal Rules then may look like this (The “quotes” in this example are used to show the contents of a string.):

IF (Road.classification EQ “country road” AND Scale (<=20000)) THEN drawCurve (“N50_specification.Solid_thin_red_line”, Road.segment)

Here Scale is a function that gets the display scale from the display device.

The portrayal rule shall refer to the appropriate attributes, functions and relationships defined in an application schema. The portrayal rule set shall also list the contexts used, including their data types.

EXAMPLE 3 In these cases external functions are necessary:

- The electronic map in a car navigation system has to be displayed so that the up-direction of the map is always in the direction the car is moving. To be able to specify the rotation of the map, the current position of the car must be retrieved continuously from an external position device using an external function.

- For electronic chart displays onboard a vessel some of the symbols are only valid for certain scale-intervals. To be able to turn the symbols on and off the system must be told what scale the map is displayed in by the display part of the chart system. A danger zone is defined spatially as a surface. Below a certain scale the danger zone is better displayed by a point symbol. An external function may be used to compute the centroid of the area and the coordinates of the centroid used to position the point symbol.

- An external function may be used to avoid visual conflicts between text and symbols placed on a map, or to handle the placement of text along curves.

7.14 Portrayal Catalogues

TBD

8 Package – ISO 19117 Portrayal [revision]

8.1 Package semantics

Portrayal is the presentation of information to humans. In the context of geographic information this has typically been accomplished by means of paper maps and more recently by means of online map displays. The portrayal process starts with geographic data, assigns symbols to the individual features, and renders the symbolized features on a graphic medium. Portrayal is a combination of data and processes. This standard focuses on the data structures which support the portrayal process, in particular the data structures for mapping features to symbols and the data structures describing symbols and their organization in collections.

A portrayal symbol is a graphic representation of a feature instance. It represents the type and possibly attributes of a feature. To represent a feature and its attributes a symbol uses graphic elements. Some of these graphic elements have meaning in their own right while others carry their meaning only within the context of a symbol. A symbol has meaning and is placed as a whole in the portrayal. A symbol element only has meaning within the context of a symbol and is placed in the portrayal only as part of a symbol. The relative position of the symbol elements is significant to the meaning of the symbol. If the relative position of the symbol elements is not maintained the symbol loses its meaning (Figure 9).

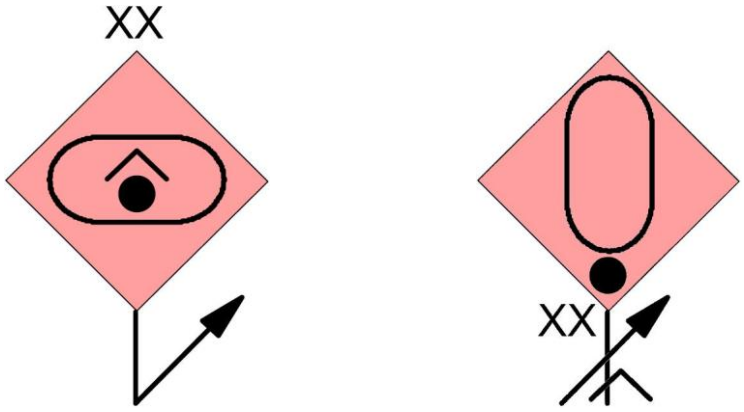

Figure 9 — Symbol elements with and without meaning

8.2 Package structure

This standard presents a conceptual schema for describing portrayal rule sets, symbols, and symbol collections, all intended for use in the portrayal of geospatial data. The conceptual schema is specified using the Unified Modelling Language (UML) [ISO 19501], following the guidance of ISO/TS 19103.

The schema is contained in three UML packages: Schema Mapping, Portrayal Feature, and Presentation. The Schema Mapping package is a conceptual schema for defining the transformation of data from one application schema to another. The Portrayal Feature package specializes the Schema Mapping and Feature Cataloguing packages for portrayal. The Presentation package defines a root for presentation and several specializations. Classes in the packages of this schema are derived from classes included in this schema as well as classes included in the packages Feature Cataloguing [ISO 19110] and Catalogues [ISO 19139]. Classes in the packages of this schema reference and use as data types classes included in this schema as well as classes included in the packages Feature Cataloguing [ISO 19110], Basic Types [ISO/TS 19103], Geometry [ISO 19107], Coordinate Reference Systems [ISO 19111], Citation and responsible party information [ISO 19115], Reference system information [ISO 19115], and Feature Data Model [ISO 19133].

9 Package – Schema Mapping

9.1 Package semantics

An application schema defines the structure of geographic data required by one or more applications. Conversely an application requires data conforming to a given application schema. Data that is not structured conformant to the application schema must be transformed or it is not usable by the application. The original data conforms to a source application schema while the transformed data conforms to a destination application schema. The schema mapping package is a conceptual schema for defining this type of transformation.

A schema mapping is equivalent to the set theoretical concept of a function. A function from a set *A* into a set *B* is defined as a rule that assigns to each element $a \in A$ a unique element $b \in B$ [Gill76]. The set *A* is called the domain of the function while the set *B* is called the codomain. The subset of the codomain *B* that is assigned to from the domain *A* by the function *f* is called the range of *f* (Figure 10).

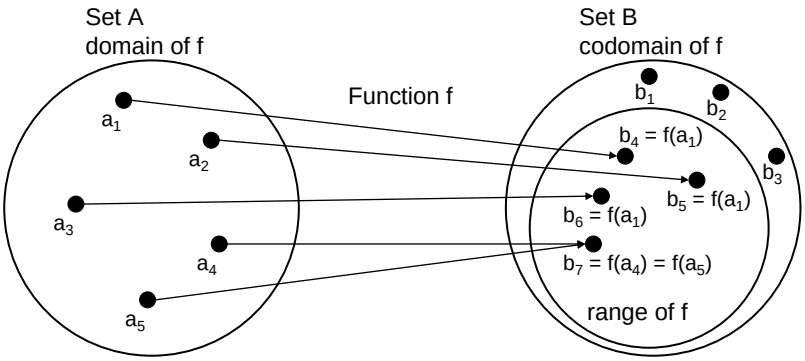

Figure 10: Mapping from set A to set B

A schema mapping may have contextual information. This is particularly important in portrayal where the portrayal context may determine symbolization. Information such as viewing conditions, medium, and rendering scale may influence symbolization and are thus part of the context. The context may determine which mapping to apply but may also be used as input to the mapping to achieve the same result. In the former case a condition attached to the mapping is used to test the mappings applicability to a context. In the latter case multiple, conceptually separate mappings are combined into one using the context to choose symbolization in the mapping. These two approaches may also be combined, using the context to select a mapping and then using the context in the mapping to choose symbolization.

This package specifies three different approaches to mapping: rule based mapping, transformation mapping, and population mapping. The rule based mapping starts with a feature and possibly a collection of subfeatures . This feature and possible subfeatures are evaluated against a collection of rules. The rules may be hierarchically structured to form a decision tree. The result of this evaluation is a new feature and possible subfeatures. The transformation mapping transforms collections of the same kind of features. This is not the entire domain, which may consist of features of many different types. The transformation mapping chains rules that select and transform collected features from the source application schema to the destination schema structure. This transformation is directed at relational databases with features collected in tables. The population mapping starts with a feature in the range (destination dataset) and populates it from the domain (source dataset).

9.2 Package structure

The schema mapping package has four subpackages (Figure 11): schema mapping root, rule based mapping, transformation mapping, and population mapping. The package is closely linked to the ISO 19110 ? Feature Catalogue, Feature Cataloguing package. All elements of these packages carry the prefix "MA_". The schema mapping root package contains the abstract root class for all types of mapping.

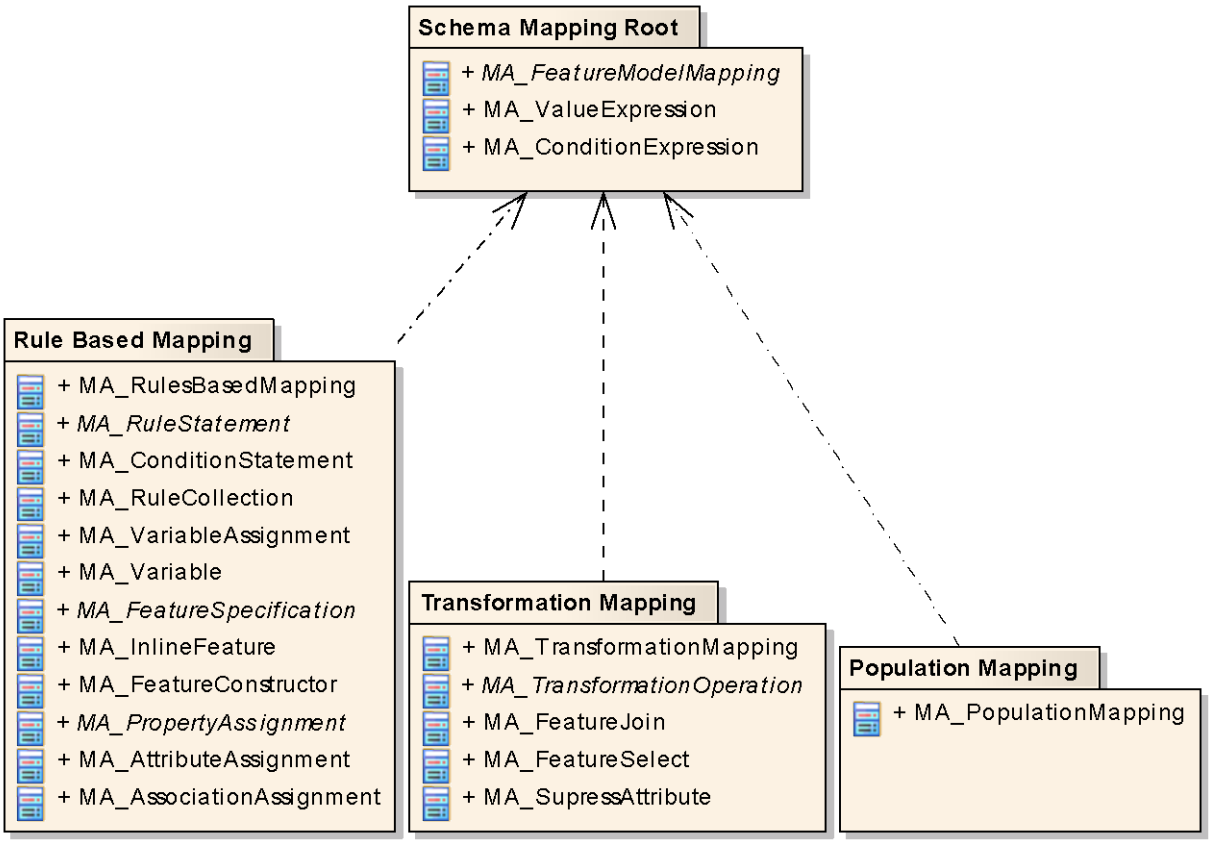

Figure 11 — Package Structure

9.3 Package – Schema Mapping Root

9.3.1 Package semantics

The Schema Mapping Root package is the root package for schema mapping. It contains three types. MA_FeatureModelMapping is the abstract root class of the schema mapping inheritance hierarchy and maps one application schema to another. The two types MA_ValueExpression and MA_ConditionExpression are expressions that define a value. The value of MA_ConditionExpression is a Boolean and is used to test if a MA_FeatureModelMapping is appropriate for a given context.

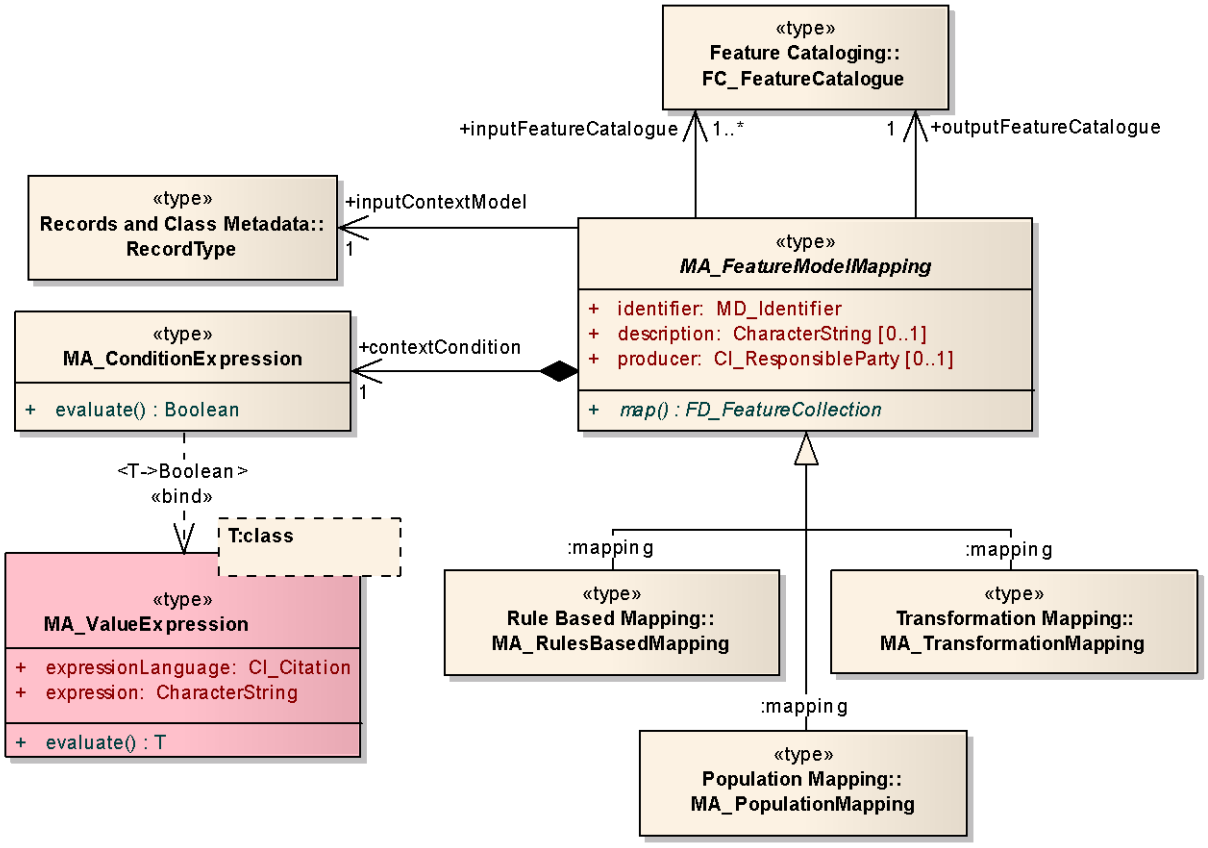

Figure 12 — Schema Mapping

9.3.2 Type – MA_FeatureModelMapping

9.3.2.1 Class semantics

MA_FeatureModelMapping is the abstract root type describing a feature model mapping, which maps one feature model (i.e. application schema) to another. The mapping can be context dependent. A record type specifies the context parameters. A condition expression is used to test a context instance. If the condition is "True" then the mapping is appropriate for the context.

The generalization set "mapping" specifies the mapping aspect of a MA_FeatureModelMapping. The types in this generalization set specialize the type of mapping (i.e. rule based, transformation, population).

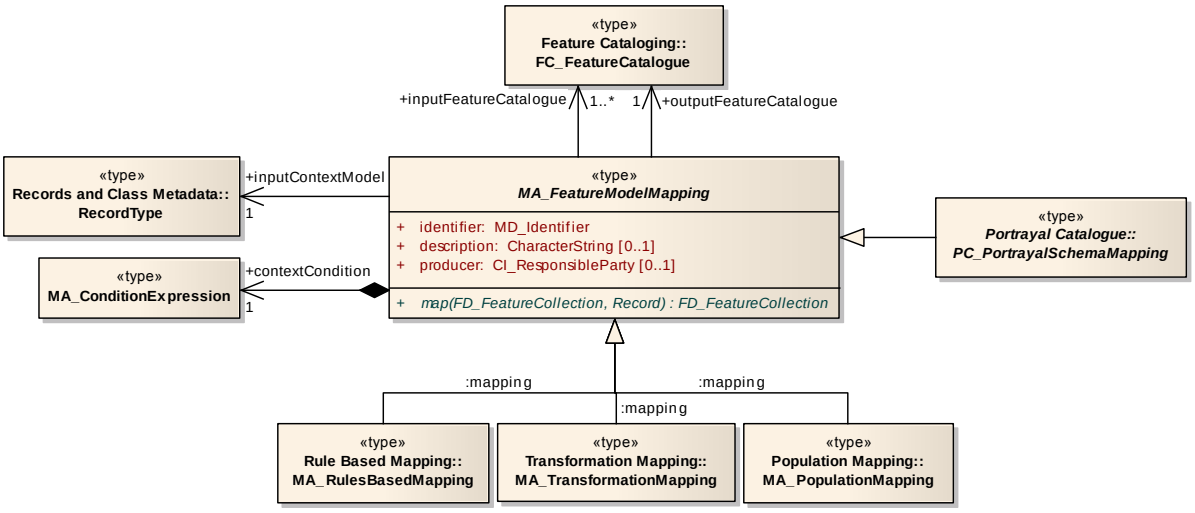

Figure 13 — MA_FeatureModelMapping context diagram

9.3.2.2 Aggregation role – contextCondition

The composition role "contextCondition" specifies a Boolean expression, which, if it evaluates to true with a given context, indicates that the schema mapping is appropriate for that context.

```
MA_FeatureModelMapping::contextCondition[1] : MA_ConditionExpression
```

9.3.2.3 Association role – inputContextModel

The association role "inputContextModel" references the record type that is used for the mapping context. This may be shared by multiple mappings that have mutually exclusive context condition expressions.

```
MA_FeatureModelMapping::inputContextModel[1] : RecordType
```

9.3.2.4 Association role – inputFeatureCatalogue

The association role "inputFeatureCatalogue" references the feature catalogue, which defines the input feature collection data structures. The multiplicity is for complementary feature catalogues (e.g. ENC, TOD), not for competing ones (e.g. ENC, DNC).

```
MA_FeatureModelMapping::inputFeatureCatalogue[1..*] :
     FC_FeatureCatalogue
```

9.3.2.5 Association role – outputFeatureCatalogue

The association role "outputFeatureCatalogue" references the feature catalogue, which defines the output feature collection data structures.

```
MA_FeatureModelMapping::outputFeatureCatalogue[1] : FC_FeatureCatalogue
```

9.3.2.6 Attribute – identifier

The attribute "identifier" is used to identify the mapping.

```
MA_FeatureModelMapping::identifier : MD_Identifier
```

9.3.2.7 Attribute – description

The optional attribute "description" may be used to store a description of the mapping.

```
MA_FeatureModelMapping::description : CharacterString [0..1]
```

9.3.2.8 Attribute – producer

The optional attribute "producer" identifies the producer of the mapping.

```
MA_FeatureModelMapping::producer : CI_ResponsibleParty [0..1]
```

9.3.2.9 Operation – map

The operation "map" shall accept a feature collection and a context record as input and return the feature collection resulting from the mapping.

```
MA_FeatureModelMapping::map(
     inputCollection : FD_FeatureCollection
     context : Record
   ) : FD_FeatureCollection
```

9.3.3 Type – MA_ValueExpression

9.3.3.1 Class semantics

A MA_ValueExpression is an expression in an expression language that evaluates to the template type. The value expression may contain literal values, references to a context elements, and feature attributes.

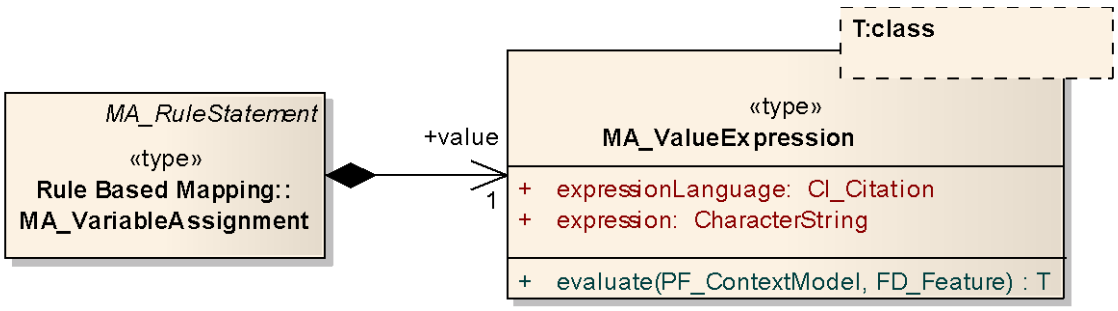

Figure 14 — MA_ValueExpression context diagram

9.3.3.2 Attribute – expressionLanguage

The attribute "expressionLanguage" cites the expression language used in the expression string.

```
MA_ValueExpression::expressionLanguage : CI_Citation
```

9.3.3.3 Attribute – expression

The attribute "expression" is the character string of the expression in the cited expression language.

```
MA_ValueExpression::expression : CharacterString
```

9.3.3.4 Operation – evaluate

The operation "evaluate" shall accept a context record and a feature as input and return the result of the expression evaluated with the specified context and feature.

```
MA_ValueExpression::evaluate(
     context : PF_ContextModel
     feature : FD_Feature
   ) : T
```

9.3.4 Type – MA_ConditionExpression

9.3.4.1 Class semantics

A MA_ConditionExpression is a Boolean expression in a given expression language. It is a Boolean binding of the MA_ValueExpression template.

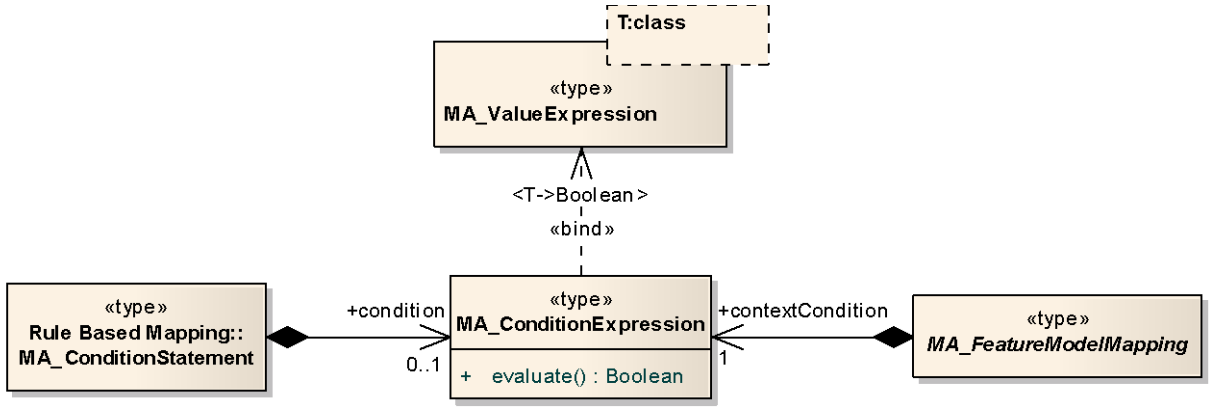

Figure 15 — MA_ConditionExpression context diagram

9.3.4.2 Operation – evaluate

The operation "evaluate" is the Boolean binding of the MA_ValueExpression:evaluate operation. It shall accept a context record and a feature as input and return the Boolean result of the expression evaluated with the specified context and feature.

```
MA_ConditionExpression::evaluate(
     context : PF_ContextModel
     feature : FD_Feature
   ) : Boolean
```

9.4 Package – Rule Based Mapping

9.4.1 Package semantics

The Rule Based Mapping package specifies a rule based schema mapping. The package defines a schema of hierarchical rules with feature specifications as leaves of the hierarchy. The package contains 12 types. MA_RulesBasedMapping is the schema mapping derived from MA_FeatureModelMapping. Five types define rule statements, MA_RuleStatement is the abstract root and the class MA_ConditionStatement, MA_RuleCollection, MA_VariableAssignment, and MA_FeatureSpecification are its derived classes. These classes define a grammar that, lacking a looping capability, is not Turing complete. MA_Variable defines the variable assigned to by MA_VariableAssignment. MA_FeatureSpecification is further specialized as MA_InlineFeature and MA_FeatureConstructor. MA_FeatureConstructor uses MA_PropertyAssignment, MA_AttributeAssignment, and MA_AssociationAssignment to assign values to a features properties.

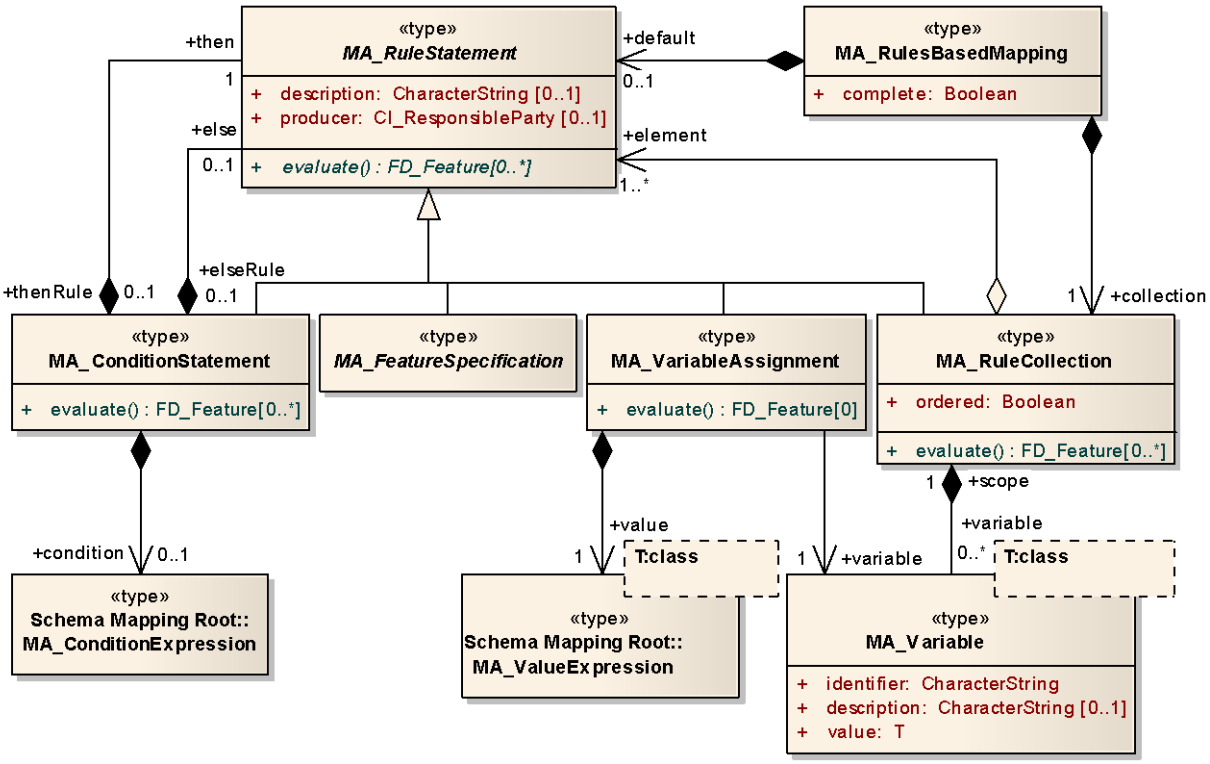

Figure 16 — Rule Statement

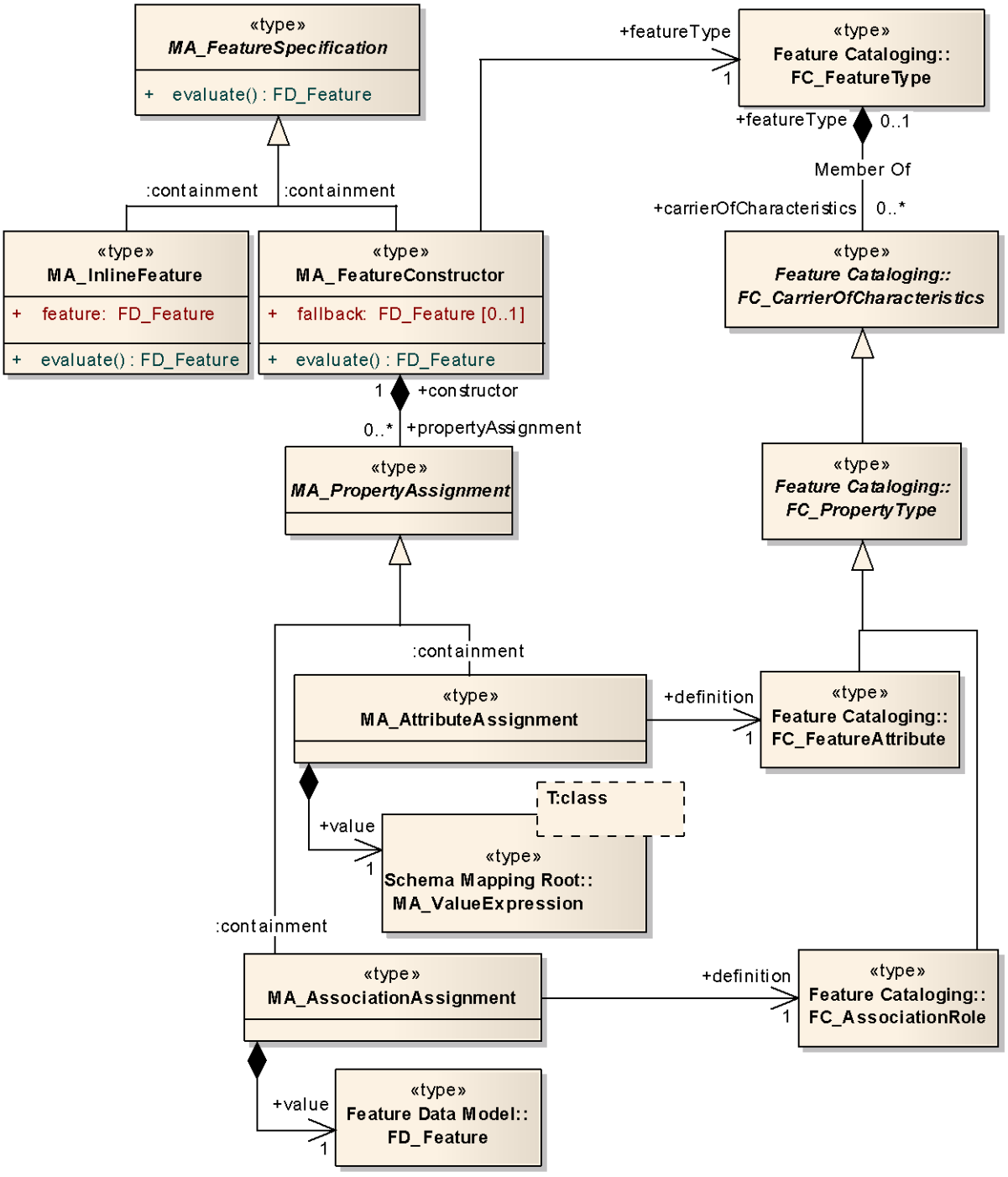

Figure 17 — Feature Specification

9.4.2 Type – MA_RulesBasedMapping

9.4.2.1 Class semantics

MA_RulesBasedMapping specializes MA_FeatureModelMapping as a rule based mapping. A rule base mapping consists of a collection of rules that evaluate to features. There may also be a default rule should the result of the rule collection be empty.

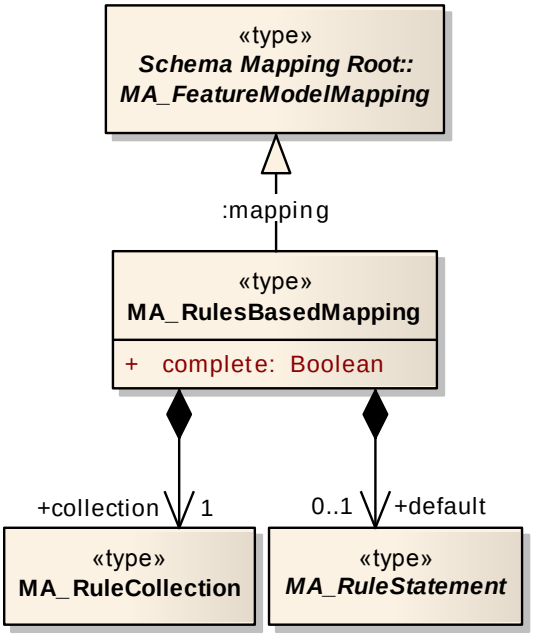

Figure 18 — MA_RulesBasedMapping context diagram

9.4.2.2 Generalization – MA_FeatureModelMapping

MA_RulesBasedMapping specializes the abstract type "MA_FeatureModelMapping". As such it shall implement all inherited attributes, operations and associations.

9.4.2.3 Aggregation role – collection

The composition role "collection" specifies the rule collection that comprises the rule based mapping.

```
MA_RulesBasedMapping::collection[1] : MA_RuleCollection
```

9.4.2.4 Aggregation role – default

The composition role "default" specifies the rule statement that should be evaluated should the rule collection evaluate to an empty set.

```
MA_RulesBasedMapping::default[0..1] : MA_RuleStatement
```

9.4.2.5 Attribute – complete

The attribute "complete" specifies if mapping is complete.

```
MA_RulesBasedMapping::complete : Boolean
```

9.4.3 Type – MA_RuleStatement

9.4.3.1 Class semantics

MA_RuleStatement is the abstract root type for schema mapping rules. Through rule collections (MA_RuleCollection) rules may be hierarchical. The leaves of a hierarchy are feature specifications (MA_FeatureSpecification). Variables (MA_VariableAssignment, MA_Variable) may be used to select attribute values or associated features which are then passed to a feature specification.

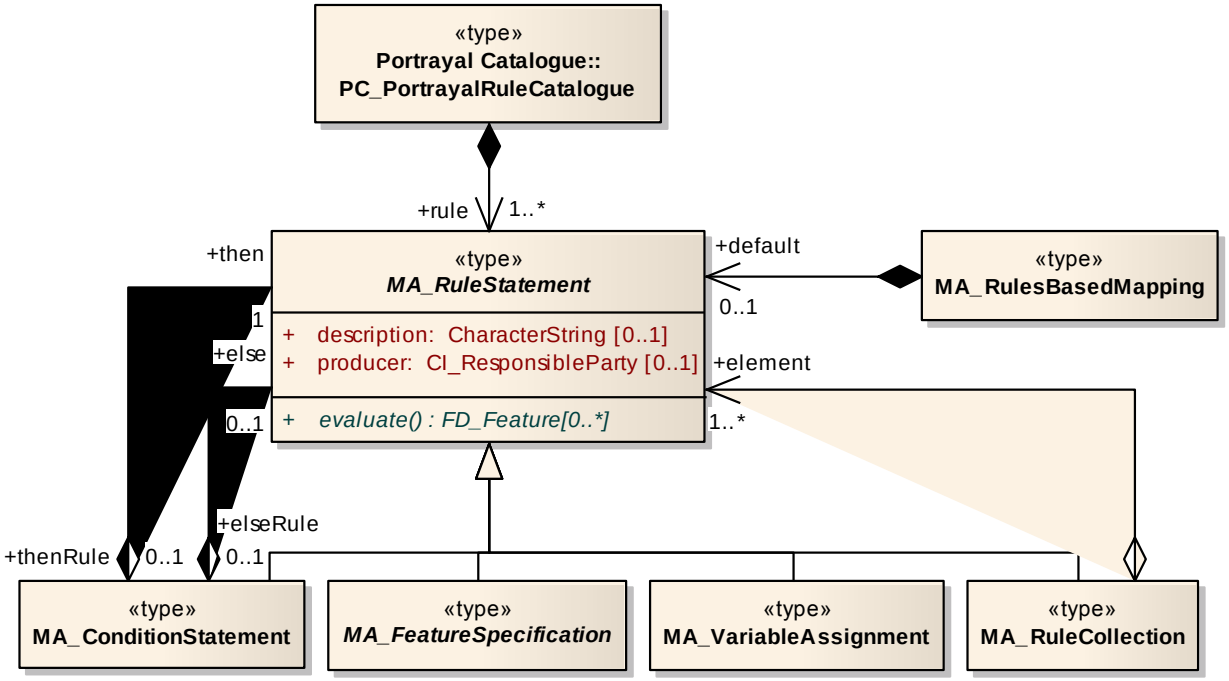

Figure 19 — MA_RuleStatement context diagram

9.4.3.2 Aggregation role – thenRule

The composition role "thenRule" specifies the referencing condition statement.

```
MA_RuleStatement::thenRule[0..1] : MA_ConditionStatement
```

9.4.3.3 Aggregation role – elseRule

The composition role "elseRule" specifies the referencing condition statement.

```
MA_RuleStatement::elseRule[0..1] : MA_ConditionStatement
```

9.4.3.4 Attribute – description

The optional attribute "description" may be used to store a description of the rule statement.

```
MA_RuleStatement::description : CharacterString  [0..1]
```

9.4.3.5 Attribute – producer

The optional attribute "producer" identifies the producer of the rule statement.

```
MA_RuleStatement::producer : CI_ResponsibleParty  [0..1]
```

9.4.3.6 Operation – evaluate

The abstract operation "evaluate" shall accept a feature and a context record as input and return a set of features.

```
MA_RuleStatement::evaluate(
     feature : FD_Feature[1..*]
     context : Record
   ) : FD_Feature[0..*]
```

9.4.4 Type – MA_ConditionStatement

9.4.4.1 Class semantics

MA_ConditionStatement specializes MA_RuleStatement as a Boolean condition and a result (then) and an optional alternative (else).

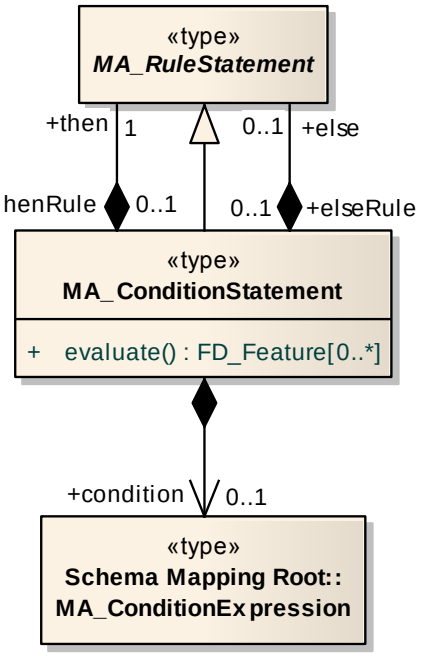

Figure 20 — MA_ConditionStatement context diagram

9.4.4.2 Generalization – MA_RuleStatement

MA_ConditionStatement specializes the abstract type "MA_RuleStatement". As such it shall implement all inherited attributes, operations and associations.

9.4.4.3 Aggregation role – then

The composition role "then" specifies the rule statement that should be evaluated if the condition of the condition statement evaluates to "True".

```
MA_ConditionStatement::then[1] : MA_RuleStatement
```

9.4.4.4 Aggregation role – else

The composition role "else" specifies the rule statement that should be evaluated if the condition of the condition statement evaluates to "False".

```
MA_ConditionStatement::else[0..1] : MA_RuleStatement
```

9.4.4.5 Aggregation role – condition

The composition role "condition" specifies the condition that is evaluated as part of the condition statement.

```
MA_ConditionStatement::condition[0..1] : MA_ConditionExpression
```

9.4.4.6 Operation – evaluate

The operation "evaluate" implements the abstract operation MA_RuleStatement::evaluate and shall accept a feature and a context record as input and return a set of features. The return value is the value of a referenced MA_RuleStatement dependent on the associated MA_ConditionExpression. If the condition expression evaluates to true the MA_RuleStatement referenced by the "then" association role is used otherwise the MA_RuleStatement reference bi the optional "else" association role is used.

```
MA_ConditionStatement::evaluate(
     feature : FD_Feature[1..*]
     context : Record
   ) : FD_Feature[0..*]
```

9.4.5 Type – MA_RuleCollection

9.4.5.1 Class semantics

MA_RuleCollection specializes MA_RuleStatement as a collection of MA_RuleStatements. MA_RuleCollection may also define the scope of variables.

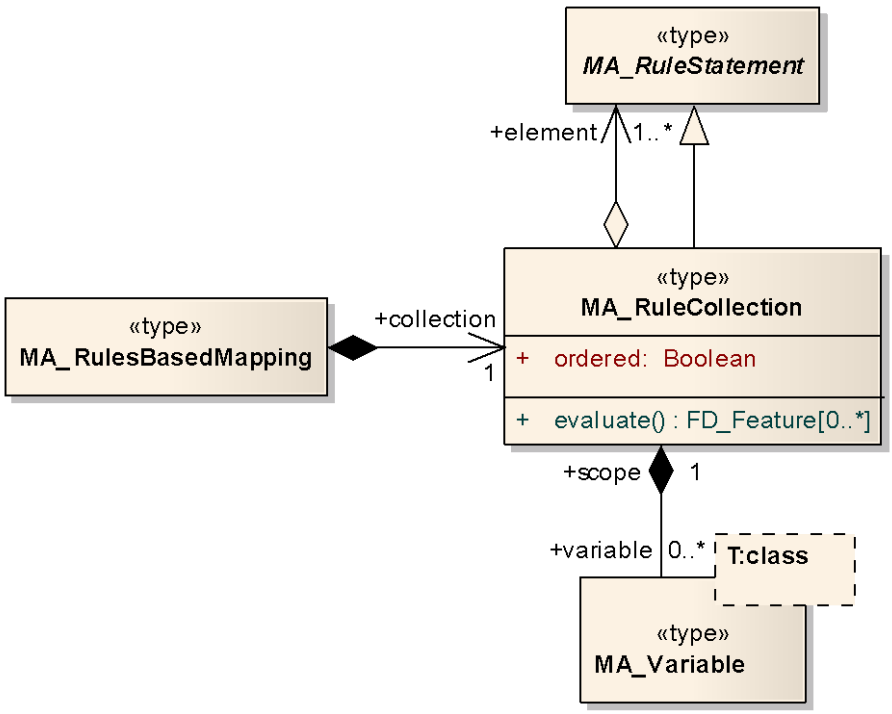

Figure 21 — MA_RuleCollection context diagram

9.4.5.2 Generalization – MA_RuleStatement

MA_RuleCollection specializes the abstract type "MA_RuleStatement". As such it shall implement all inherited attributes, operations and associations.

9.4.5.3 Aggregation role – variable

The composition role "variable" specifies the variables that are defined in the scope of the rule collection. Variables defined in enclosing collection are valid in the subordinate scope but not related by a direct association.

```
MA_RuleCollection::variable[0..*] : MA_Variable
```

9.4.5.4 Aggregation role – element

The aggregation role "element" specifies the collection of rule statements that make up a rule collection.

```
MA_RuleCollection::element[1..*] : MA_RuleStatement
```

9.4.5.5 Attribute – ordered

The attribute "ordered" specifies if the element rule statements are ordered.

```
MA_RuleCollection::ordered : Boolean
```

9.4.5.6 Operation – evaluate

The operation "evaluate" implements the abstract operation MA_RuleStatement::evaluate and shall accept a feature and a context record as input and return a set of features. The return value is derived from the component rule statement, either as the result of all components or as the result of the first in the ordered list to return a value; this is implementation dependent.

```
MA_RuleCollection::evaluate(
     feature : FD_Feature[1..*]
     context : Record
   ) : FD_Feature[0..*]
```

9.4.6 Type – MA_VariableAssignment

9.4.6.1 Class semantics

MA_VariableAssignment specializes MA_RuleStatement. A MA_VariableAssignment assigns a value to an MA_Variable.

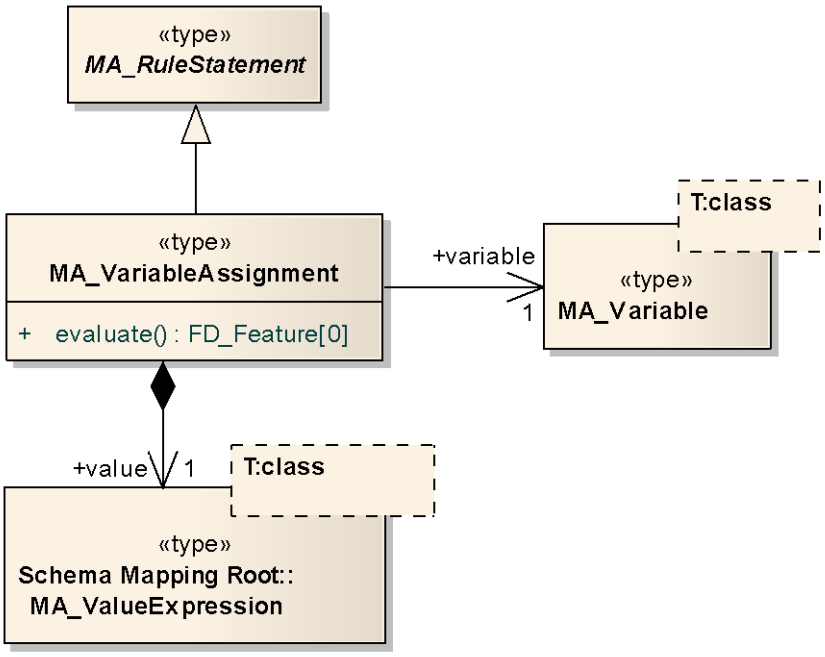

Figure 22 — MA_VariableAssignment context diagram

9.4.6.2 Generalization – MA_RuleStatement

MA_VariableAssignment specializes the abstract type "MA_RuleStatement". As such it shall implement all inherited attributes, operations and associations constraining the multiplicity of the return value of the "evaluate" operation to FD_Feature[0].

9.4.6.3 Aggregation role – value

The composition role "value" specifies the value expression, which is the source of the value to be assigned to a variable.

```
MA_VariableAssignment::value[1] : MA_ValueExpression
```

9.4.6.4 Association role – variable

The association role "value" specifies the variable to which a value is assigned.

```
MA_VariableAssignment::variable[1] : MA_Variable
```

9.4.6.5 Operation – evaluate

The operation "evaluate" implements the abstract operation MA_RuleStatement::evaluate and shall accept a feature and a context record as input and return an empty set of features. The effect of this operation is to assign a value to a variable.

```
MA_VariableAssignment::evaluate(
     feature : FD_Feature[1..*]
     context : Record
   ) : FD_Feature[0]
```

9.4.7 Type – MA_Variable

9.4.7.1 Class semantics

MA_Variable holds the value of a variable. A variable is assigned a value by a MA_VariableAssignment and has a scope limited by a MA_RuleCollection.

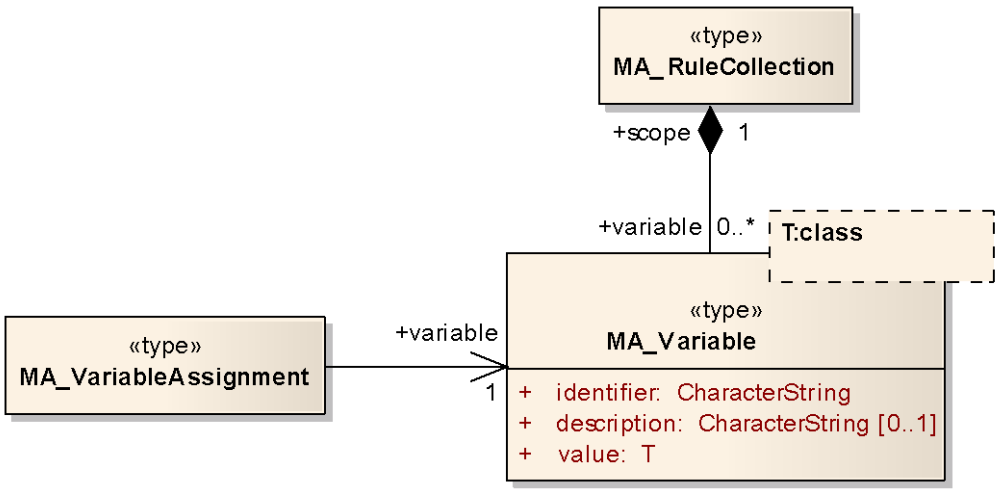

Figure 23 — MA_Variable context diagram

9.4.7.2 Aggregation role – scope

The composition role "scope" specifies the scope in which a variable is defined. The variable is valid in subordinate collection but not associated by a scope association.

```
MA_Variable::scope[1] : MA_RuleCollection
```

9.4.7.3 Attribute – identifier

The attribute "identifier" is used to identify the variable.

```
MA_Variable::identifier : CharacterString
```

9.4.7.4 Attribute – description

The optional attribute "description" may be used to store a description of the variable.

```
MA_Variable::description : CharacterString  [0..1]
```

9.4.7.5 Attribute – value

The attribute "value" is used to store the variable value.

```
MA_Variable::value : T
```

9.4.8 Type – MA_FeatureSpecification

9.4.8.1 Class semantics

MA_FeatureSpecification specializes MA_RuleStatement as an abstract specification of a feature. A MA_FeatureSpecification is further specialized as either an inline feature instance or a feature constructor.

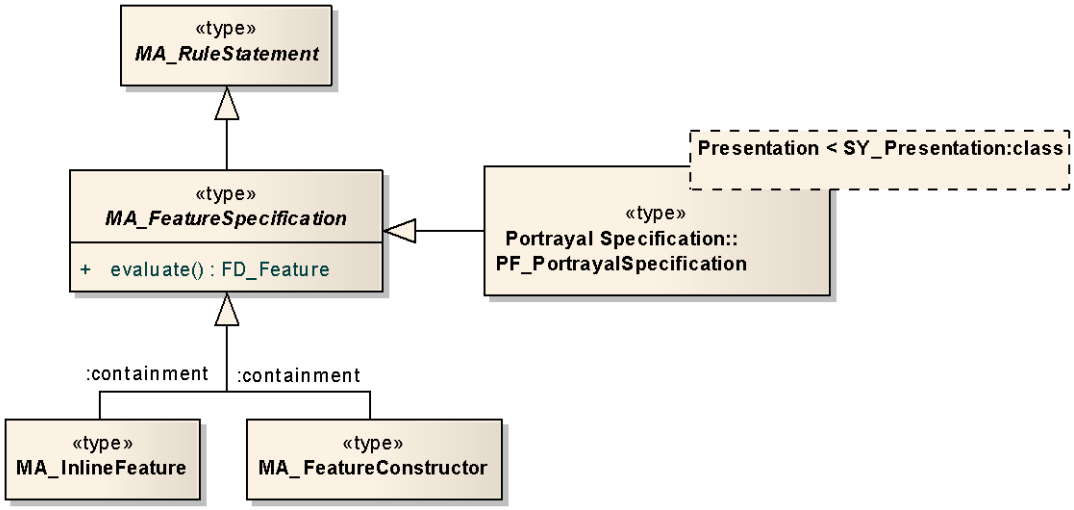

Figure 24 — MA_FeatureSpecification context diagram

9.4.8.2 Generalization – MA_RuleStatement

The abstract type "MA_FeatureSpecification" specializes the abstract type "MA_RuleStatement". As such it shall implement all inherited attributes, operations and associations constraining the multiplicity of the return value of the "evaluate" operation to one FD_Feature.

9.4.8.3 Operation – evaluate

The abstract operation "evaluate" shall accept a feature and a context record as input and return a feature.

```
MA_FeatureSpecification::evaluate(
     feature : FD_Feature[1..*]
     context : Record
   ) : FD_Feature
```

9.4.9 Type – MA_InlineFeature

9.4.9.1 Class semantics

MA_InlineFeature specializes MA_FeatureSpecification by containing a feature instance, which is returned by the "evaluate" operation.

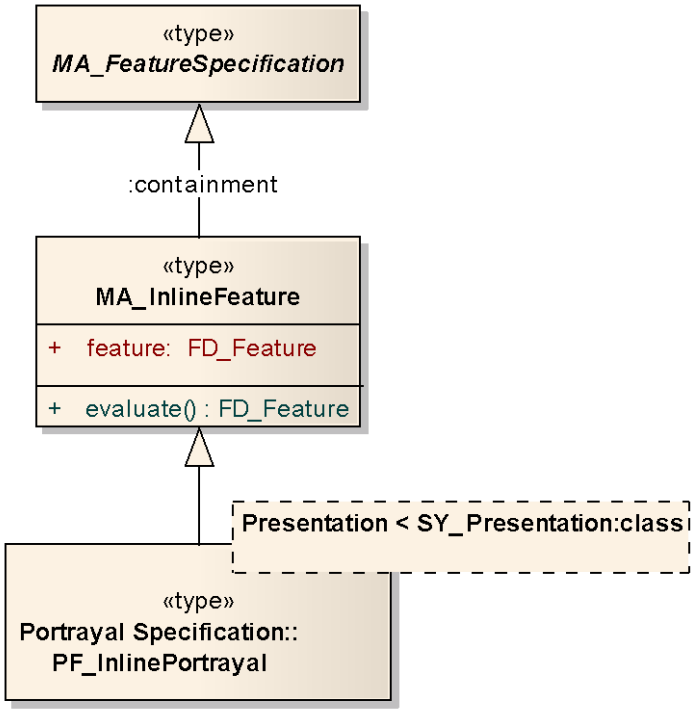

Figure 25 — MA_InlineFeature context diagram

9.4.9.2 Generalization – MA_FeatureSpecification

MA_InlineFeature specializes the abstract type "MA_FeatureSpecification" as part of the "containment" generalization set. It reimplements the "evaluate" operation with no parameters.

9.4.9.3 Attribute – feature

The attribute "feature" is the feature instance returned by the "evaluate" operation.

```
MA_InlineFeature::feature : FD_Feature
```

9.4.9.4 Operation – evaluate

The operation "evaluate" implements the abstract operation MA_FeatureSpecification::evaluate; it returns a feature.

```
MA_InlineFeature::evaluate(
  ) : FD_Feature
```

9.4.10 Type – MA_FeatureConstructor

9.4.10.1 Class semantics

MA_FeatureConstructor specializes MA_FeatureSpecification. MA_FeatureConstructor specifies the how to create a feature from input data.

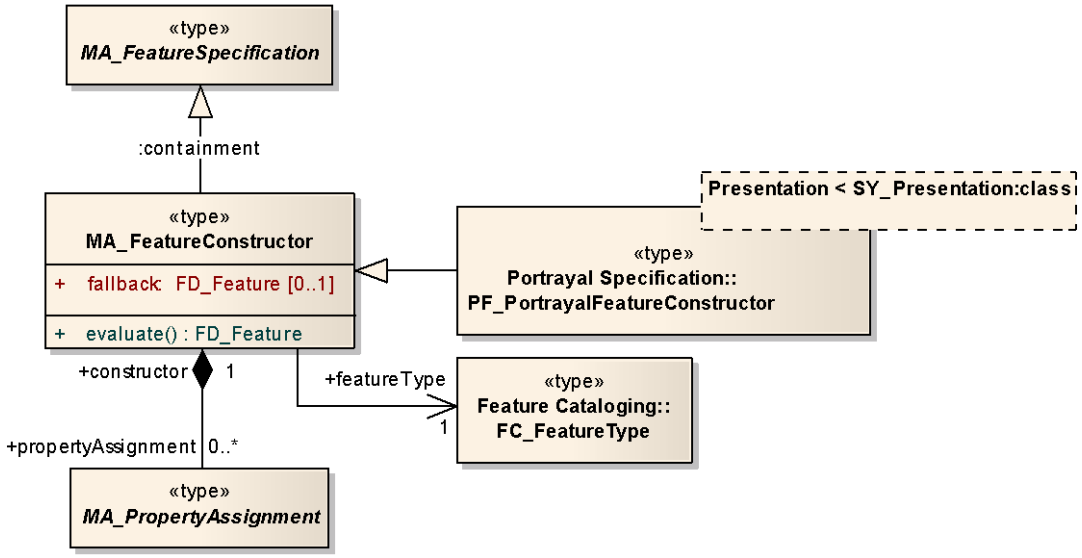

Figure 26 — MA_FeatureConstructor context diagram

9.4.10.2 Generalization – MA_FeatureSpecification

MA_FeatureConstructor specialize the abstract type "MA_FeatureSpecification" as part of the "containment" generalization set. As such it shall implement all inherited attributes, operations and associations.

9.4.10.3 Association role – featureType

The association role "featureType" specifies the feature type of the feature that is created.

```
MA_FeatureConstructor::featureType[1] : FC_FeatureType
```

9.4.10.4 Aggregation role – propertyAssignment

The composition role "propertyAssignment" specifies the property assignments used to populate a newly created feature.

```
MA_FeatureConstructor::propertyAssignment[0..*] : MA_PropertyAssignment
```

9.4.10.5 Attribute – fallback

The optional attribute "fallback" is a feature instance returned by the "evaluate" operation should construction of the feature fail.

```
MA_FeatureConstructor::fallback : FD_Feature  [0..1]
```

9.4.10.6 Operation – evaluate

The operation "evaluate" shall accept a feature and a context record as input and return a feature as the result of the constructor.

```
MA_FeatureConstructor::evaluate(
     feature : FD_Feature[1..*]
     context : Record
   ) : FD_Feature
```

9.4.11 Type – MA_PropertyAssignment

9.4.11.1 Class semantics

MA_PropertyAssignment is the abstract root type property assignment for specifying a value for a feature property as part of a feature constructor.

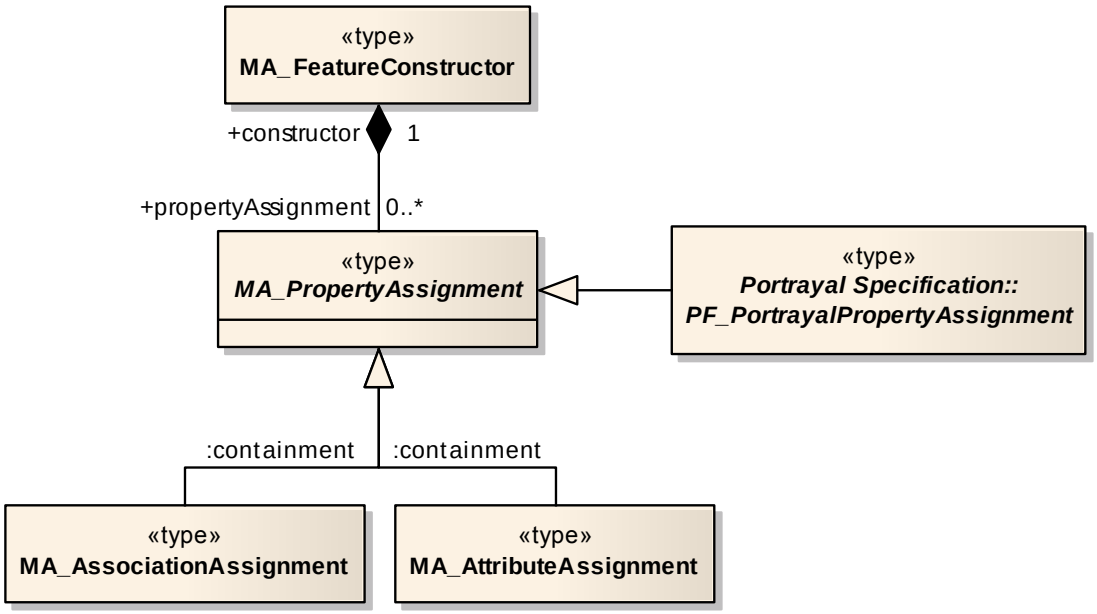

Figure 27 — MA_PropertyAssignment context diagram

9.4.11.2 Aggregation role – constructor

The composition role "constructor" references the containing feature constructor for the property assignment.

```
MA_PropertyAssignment::constructor[1] : MA_FeatureConstructor
```

9.4.12 Type – MA_AttributeAssignment

9.4.12.1 Class semantics

MA_AttributeAssignment specializes MA_PropertyAssignment. It assigns a value to an attribute of a newly created feature from a value expression.

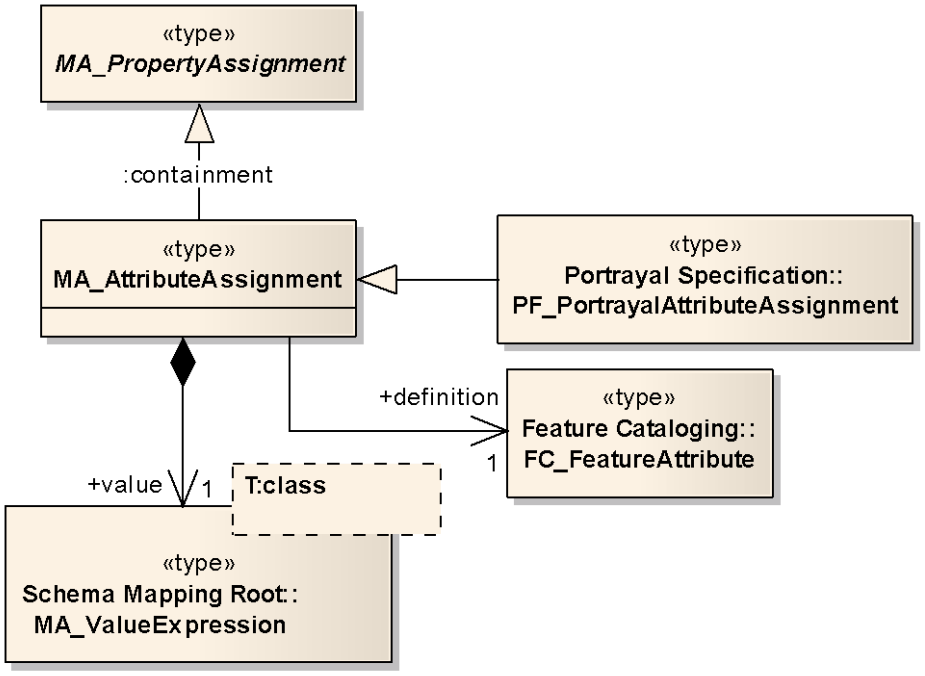

Figure 28 — MA_AttributeAssignment context diagram

9.4.12.2 Generalization – MA_PropertyAssignment

MA_AttributeAssignment specializes the abstract type "MA_PropertyAssignment" as part of the "containment" generalization set. As such it shall implement all inherited attributes, operations and associations.

9.4.12.3 Association role – definition

The association role "definition" specifies the feature attribute to which a value is assigned in the newly created feature.

```
MA_AttributeAssignment::definition[1] : FC_FeatureAttribute
```

9.4.12.4 Aggregation role – value

The composition role "value" specifies the value expression, which is the source of the value to be assigned to an attribute of the newly created feature..

```
MA_AttributeAssignment::value[1] : MA_ValueExpression
```

9.4.13 Type – MA_AssociationAssignment

9.4.13.1 Class semantics

MA_AssociationAssignment specializes MA_PropertyAssignment. It creates a relationship between a newly created feature and an existing feature.

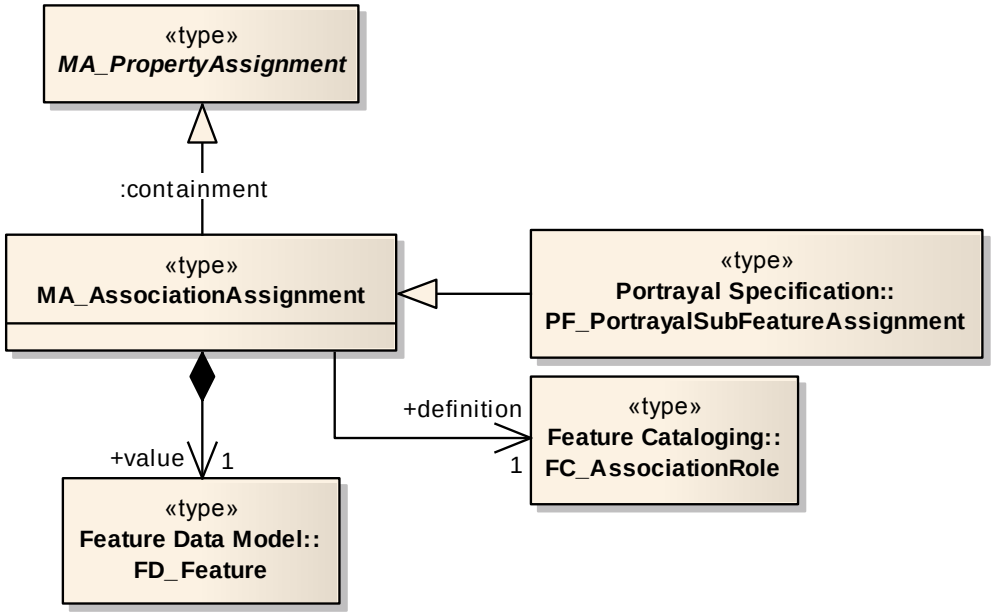

Figure 29 — MA_AssociationAssignment context diagram

9.4.13.2 Generalization – MA_PropertyAssignment

MA_AssociationAssignment specializes the abstract type "MA_PropertyAssignment" as part of the "containment" generalization set. As such it shall implement all inherited attributes, operations and associations.

9.4.13.3 Association role – definition

The association role "definition" specifies the association role to which a feature is related in the newly created feature.

```
MA_AssociationAssignment::definition[1] : FC_AssociationRole
```

9.4.13.4 Aggregation role – value

The composition role "value" specifies the feature instance, which is to be associated with the newly created feature.

```
MA_AssociationAssignment::value[1] : FD_Feature
```

9.5 Package – Transformation Mapping

9.5.1 Package semantics

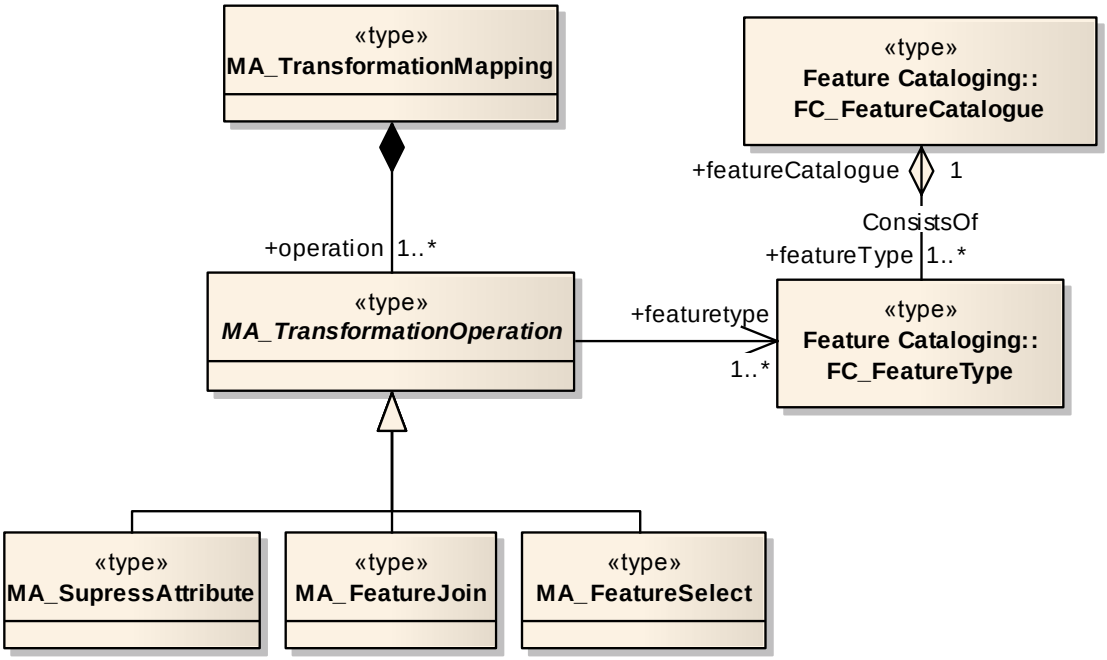

Figure 30 — Transformation Mapping

9.5.2 Type – MA_TransformationMapping

9.5.2.1 Class semantics

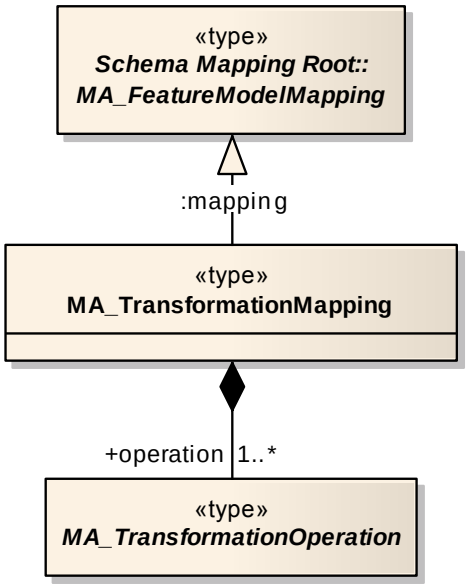

Figure 31 — MA_TransformationMapping incomplete context diagram

9.5.2.2 Generalization – MA_FeatureModelMapping

9.5.2.3 Aggregation role – operation

```
MA_TransformationMapping::operation[1..*] : MA_TransformationOperation
```

9.5.3 Type – MA_TransformationOperation

9.5.3.1 Class semantics

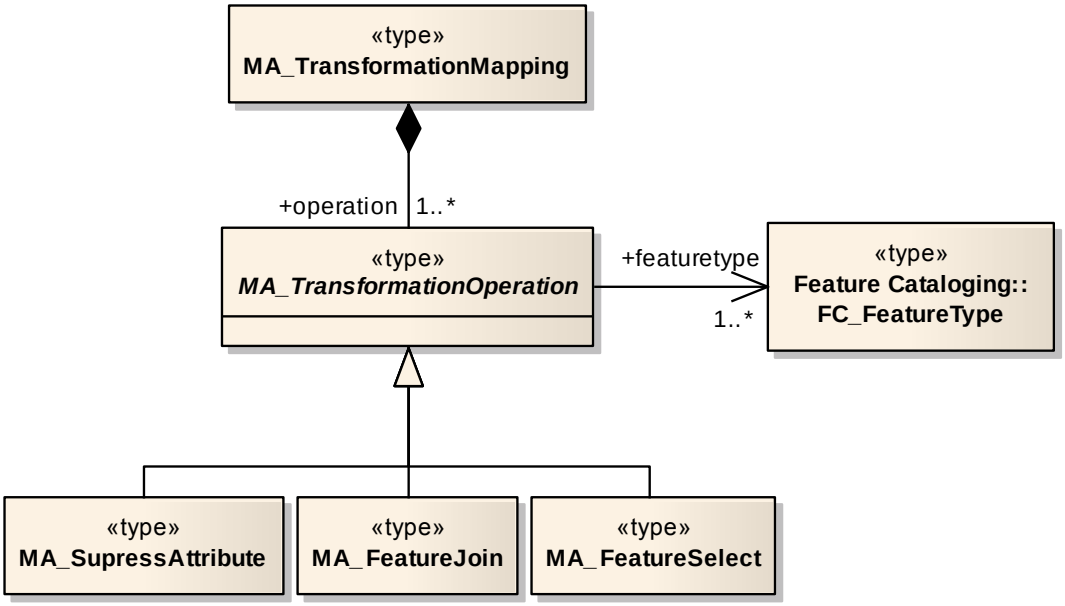

Figure 32 — MA_TransformationOperation incomplete context diagram

9.5.3.2 Association role – featuretype

```
MA_TransformationOperation::featuretype[1..*] : FC_FeatureType
```

9.5.4 Type – MA_FeatureJoin

9.5.4.1 Class semantics

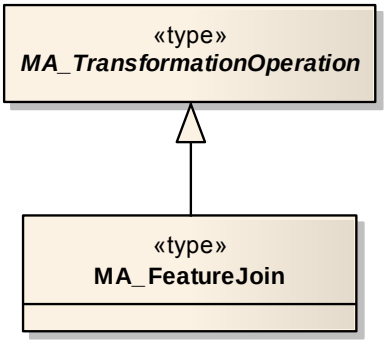

Figure 33 — MA_FeatureJoin incomplete context diagram

9.5.4.2 Generalization – MA_TransformationOperation

9.5.5 Type – MA_FeatureSelect

9.5.5.1 Class semantics

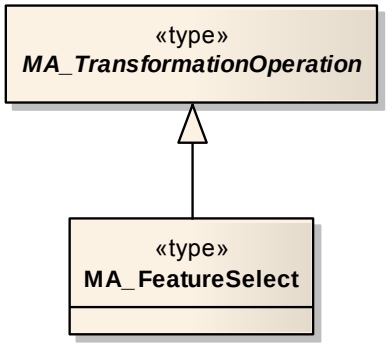

Figure 34 — MA_FeatureSelect incomplete context diagram

9.5.5.2 Generalization – MA_TransformationOperation

9.5.6 Type – MA_SupressAttribute

9.5.6.1 Class semantics

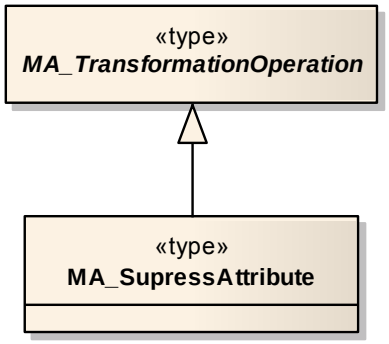

Figure 35 — MA_SupressAttribute incomplete context diagram

9.5.6.2 Generalization – MA_TransformationOperation

9.6 Package – Population Mapping

9.6.1 Package semantics

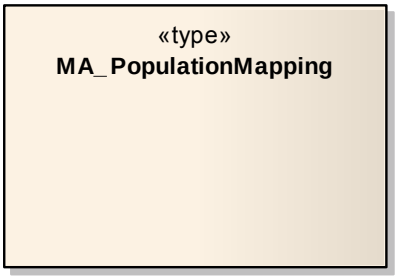

Figure 36 — Population Mapping

9.6.2 Type – MA_PopulationMapping

9.6.2.1 Class semantics

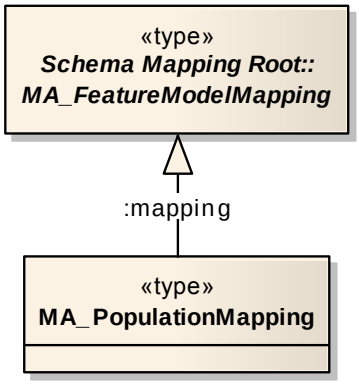

Figure 37 — MA_PopulationMapping incomplete context diagram

9.6.2.2 Generalization – MA_FeatureModelMapping

10 Package – Portrayal Feature

10.1 Package semantics

A feature is an abstraction of real world phenomena [ISO 19101]. As such it has properties that describe characteristics of the feature. These characteristics may include geometry, associations with other feature, various metadata, etc. but also characteristics of the portrayal of the feature. The primary property of a feature for portrayal is the symbol or presentation of the feature. This presentation may have variables such as size and colour. These are also properties of the feature. This package focuses on features used for portrayal, i.e. portrayal features.

This package has three subpackages: Portrayal Catalogue, Portrayal Instance, and Portrayal Specification. The Portrayal Catalogue package contains the specialization of portrayal features from the feature cataloguing schema. The Portrayal Instance package specializes instances of portrayal features from

the general feature model. The Portrayal Specification package specializes the feature specification aspect of the Rule Based Mapping.

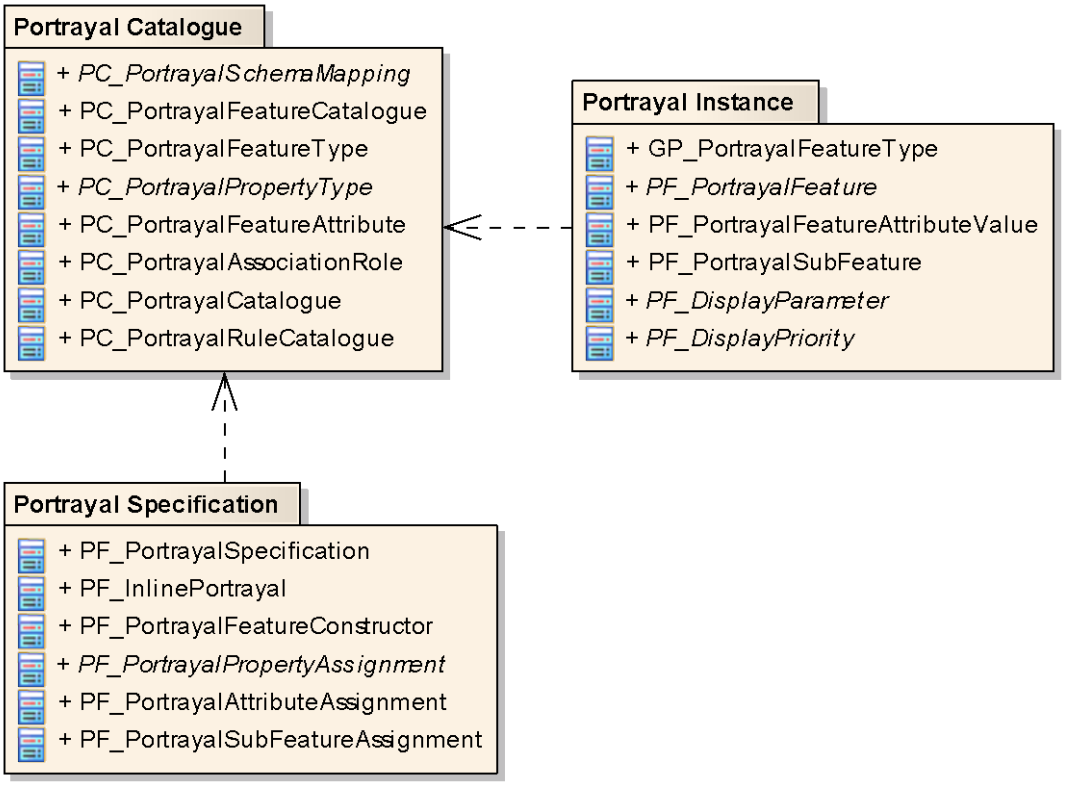

Figure 38 — Package Overview

10.2 Package – Portrayal Catalogue

10.2.1 Package semantics

The Portrayal Catalogue package specifies three aspects of portrayal catalogues. First, it specializes the mapping of feature catalogues to portrayal specific feature catalogues. Second, it specializes portrayal feature types from feature types. Third, it specializes portrayal catalogues and subcatalogues from CT_Catalogue. The package contains 8 types. The two types PC_PortrayalSchemaMapping and PC_PortrayalFeatureCatalogue specialize feature model mapping for portrayal, where a portrayal feature catalogue is a catalogue of portrayal features. The four types PC_PortrayalFeatureType, PC_PortrayalPropertyType, PC_PortrayalFeatureAttribute, and PC_PortrayalAssociationRole specialize elements of feature cataloguing for portrayal, in particular adding a presentation to all portrayal features and adding default values to properties. The two types PC_PortrayalRuleCatalogue and PC_PortrayalCatalogue are containers for portrayal feature mapping rules and the collection of all elements needed for a portrayal respectively.

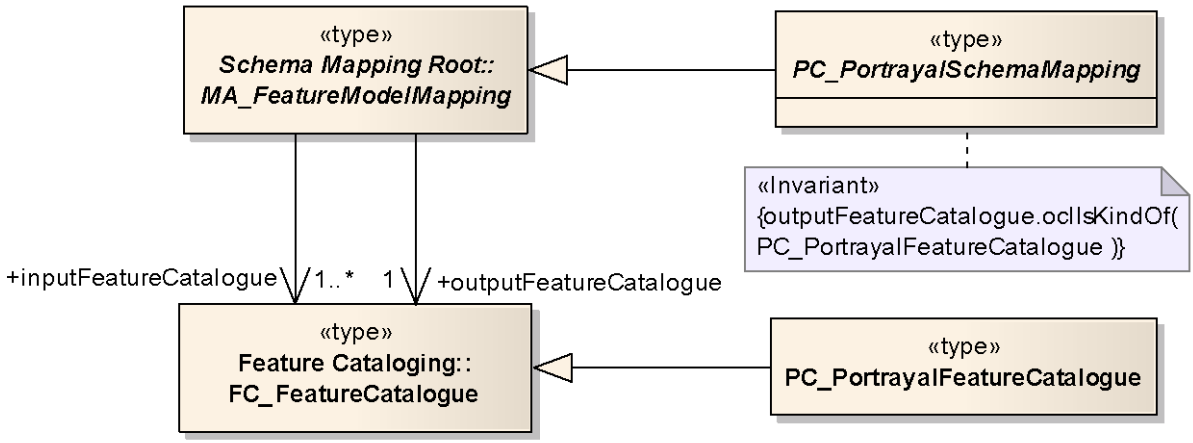

Figure 39 — Portrayal Schema Mapping

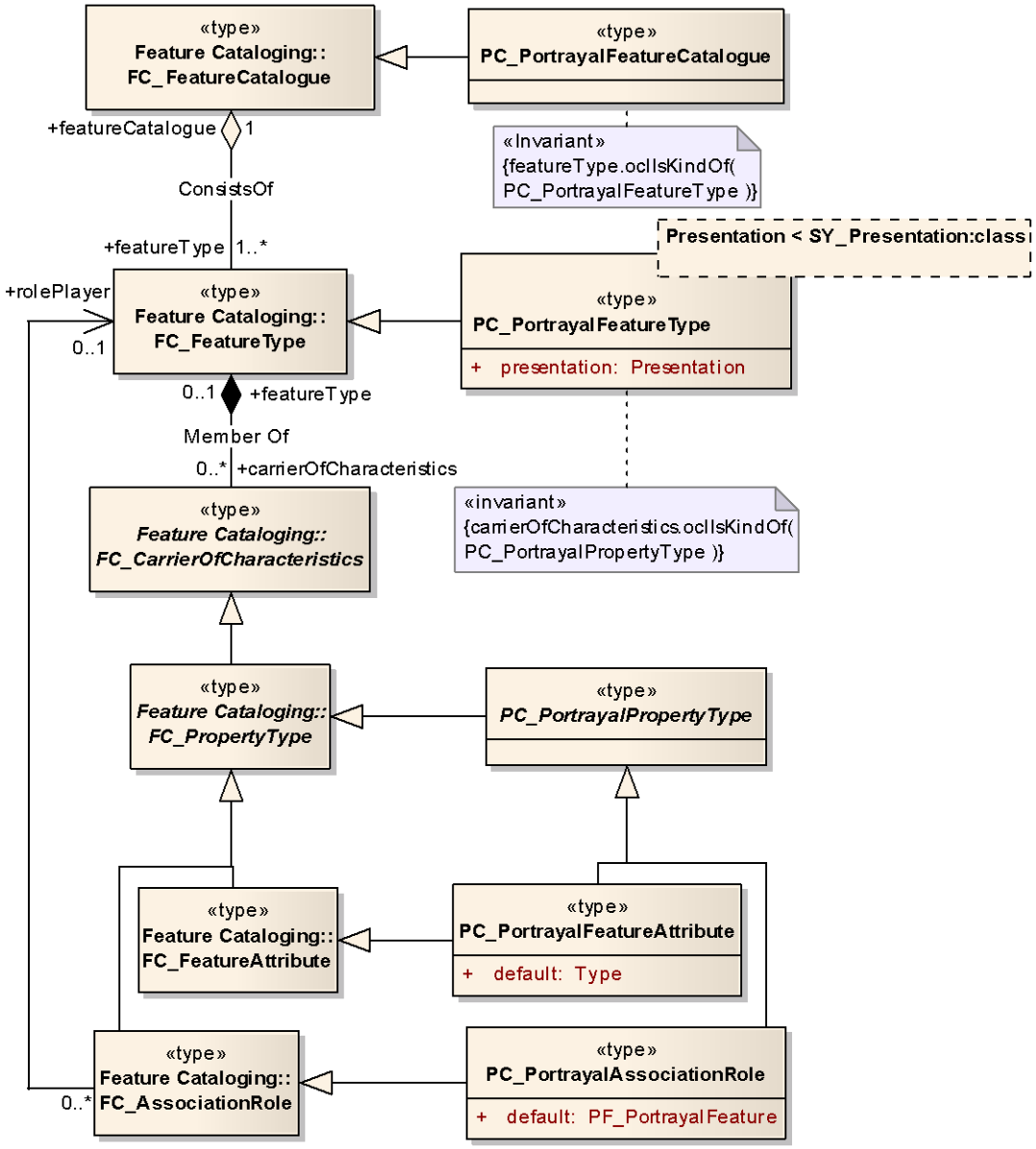

Figure 40 — Portrayal Feature Type

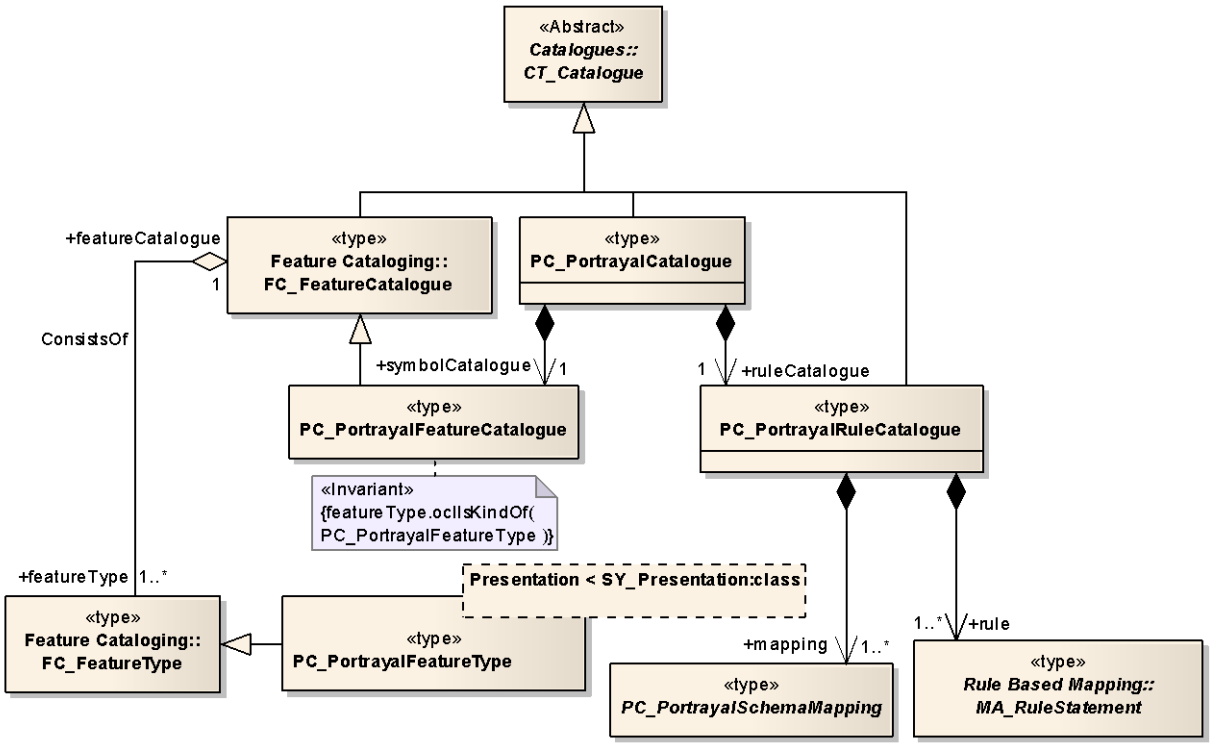

Figure 41 — Portrayal Catalogue

10.2.2 Type – PC_PortrayalSchemaMapping

10.2.2.1 Class semantics

PC_PortrayalSchemaMapping is an abstract type describing a portrayal feature model mapping. It specializes MA_FeatureModelMapping for portrayal with the association role "outputFeatureCatalogue" constrained to a "PC_PortrayalFeatureCatalogue" type. PC_PortrayalSchemaMapping combined with a type from the "mapping" generalization set of MA_FeatureModelMapping specifies a portrayal feature model mapping.

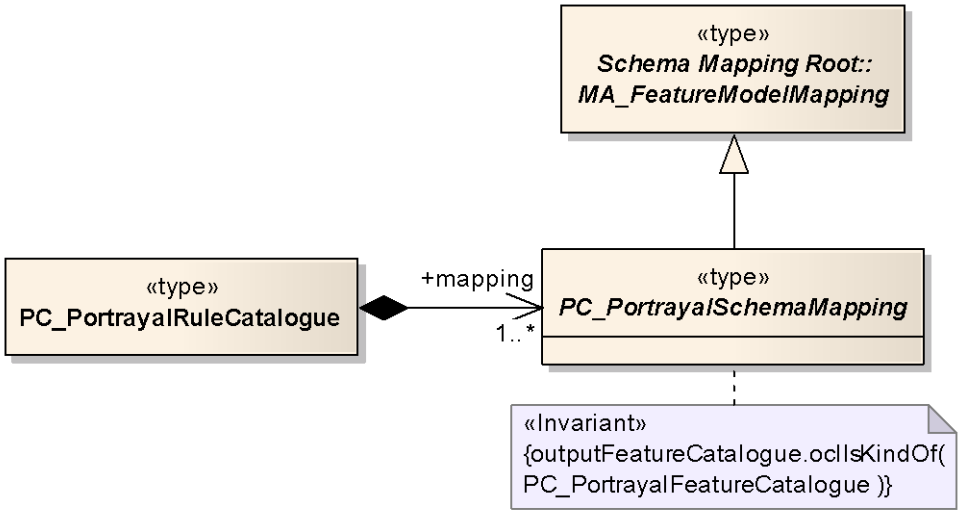

Figure 42 — PC_PortrayalSchemaMapping context diagram

10.2.2.2 Generalization – MA_FeatureModelMapping

PC_PortrayalSchemaMapping specializes the abstract type "MA_FeatureModelMapping" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.2.3 Type – PC_PortrayalFeatureCatalogue

10.2.3.1 Class semantics

A PC_PortrayalFeatureCatalogue contains the definition of a number of portrayal feature types. It specializes FC_FeatureCatalogue for portrayal with the association role "featureType" constrained to elements of the type "PC_PortrayalFeatureType".

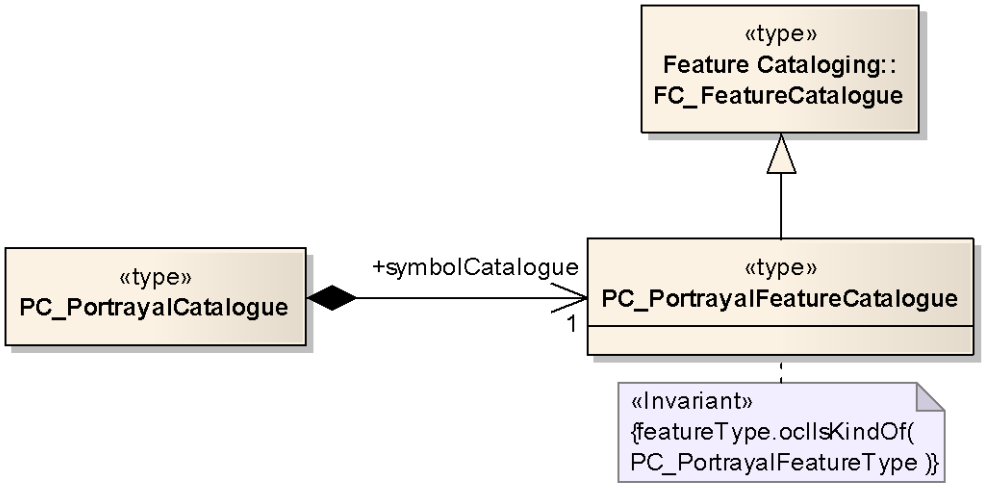

Figure 43 — PC_PortrayalFeatureCatalogue context diagram

10.2.3.2 Generalization – FC_FeatureCatalogue

PC_PortrayalFeatureCatalogue specializes the type "FC_FeatureCatalogue" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.2.4 Type – PC_PortrayalFeatureType

10.2.4.1 Class semantics

PC_PortrayalFeatureType defines a feature type with portrayal specific properties. It specializes FC_FeatureType for portrayal with the association role "carrierOfCharacteristics" constrained to elements of type "PC_PortrayalPropertyType". A portrayal feature type has a presentation which is templatized on a specialization of SY_Presentation.

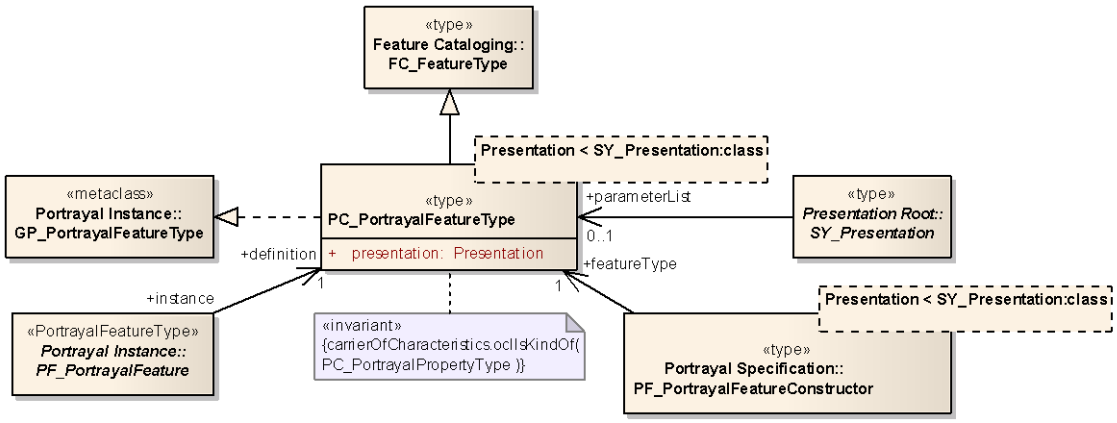

Figure 44 — PC_PortrayalFeatureType context diagram

10.2.4.2 Generalization – FC_FeatureType

PC_PortrayalFeatureType specializes the type "FC_FeatureType" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.2.4.3 Attribute – presentation

The attribute "presentation" specifies the presentation or symbol associated with a portrayal feature. The template type specifies the type of the presentation and is a specialization of SY_Presentation.

```
PC_PortrayalFeatureType::presentation : Presentation
```

10.2.5 Type – PC_PortrayalPropertyType

10.2.5.1 Class semantics

PC_PortrayalPropertyType is an abstract type for portrayal feature properties specializing FC_PropertyType.

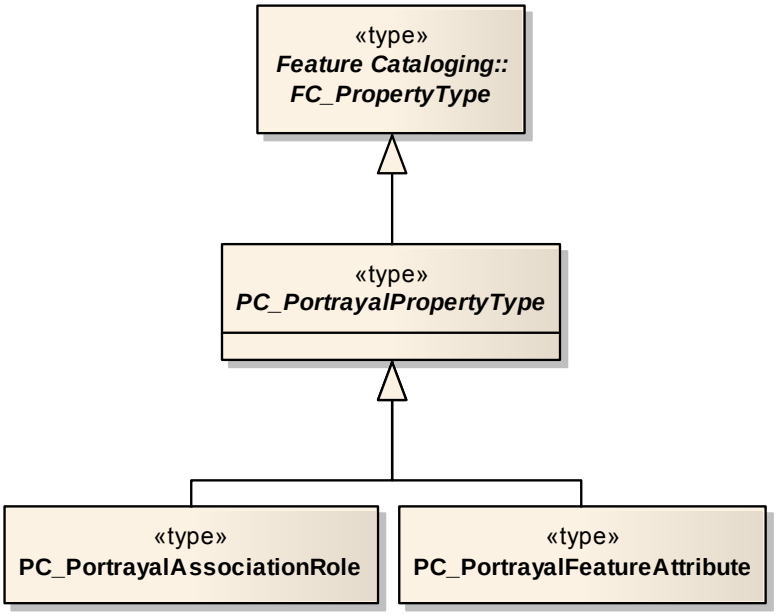

Figure 45 — PC_PortrayalPropertyType context diagram

10.2.5.2 Generalization – FC_PropertyType

PC_PortrayalPropertyType specializes the type "FC_PropertyType" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.2.6 Type – PC_PortrayalFeatureAttribute

10.2.6.1 Class semantics

PC_PortrayalFeatureAttribute specifies characteristics of a portrayal feature type. PC_PortrayalFeatureAttribute type specializes FC_FeatureAttribute for portrayal by adding a default value. It also specializes PC_PortrayalPropertyType as FC_FeatureAttribute specializes FC_PropertyType.

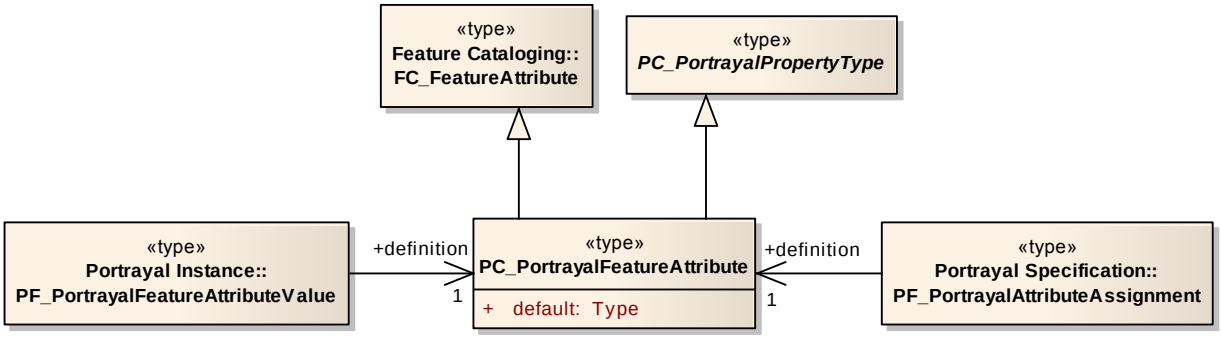

Figure 46 — PC_PortrayalFeatureAttribute context diagram

10.2.6.2 Generalization – PC_PortrayalPropertyType

PC_PortrayalFeatureAttribute specializes the type "PC_PortrayalPropertyType". As such it shall implement all inherited attributes, operations and associations.

10.2.6.3 Generalization – FC_FeatureAttribute

PC_PortrayalFeatureAttribute specializes the type "FC_FeatureAttribute" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.2.6.4 Attribute – default

The attribute "default" specifies the default value for an instance of a portrayal feature attribute.

```
PC_PortrayalFeatureAttribute::default : Type
```

10.2.7 Type – PC_PortrayalAssociationRole

10.2.7.1 Class semantics

PC_PortrayalAssociationRole specifies a role of the association FC_AssociationRole::relation for a portrayal feature type. PC_PortrayalAssociationRole type specializes FC_AssociationRole for portrayal by adding a default value. It also specializes PC_PortrayalPropertyType as FC_AssociationRole specializes FC_PropertyType.

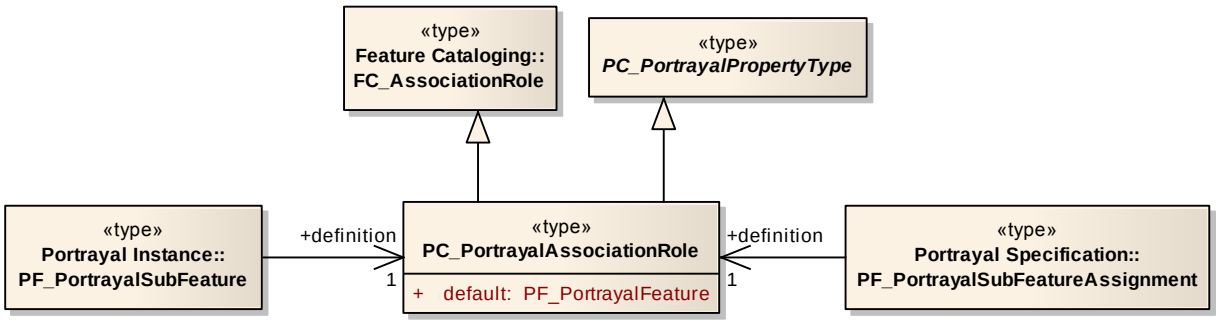

Figure 47 — PC_PortrayalAssociationRole context diagram

10.2.7.2 Generalization – FC_AssociationRole

PC_PortrayalAssociationRole specializes the type "FC_AssociationRole" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.2.7.3 Generalization – PC_PortrayalPropertyType

PC_PortrayalAssociationRole specializes the type "PC_PortrayalPropertyType". As such it shall implement all inherited attributes, operations and associations.

10.2.7.4 Attribute – default

The attribute "default" specifies the default portrayal feature with which to instantiate the portrayal feature association role.

```
PC_PortrayalAssociationRole::default : PF_PortrayalFeature
```

10.2.8 Type – PC_PortrayalCatalogue

10.2.8.1 Class semantics

PC_PortrayalCatalogue specializes CT_Catalogue containing a portrayal rule catalogue as well as a portrayal feature catalogue and as such contains the elements necessary for a portrayal.

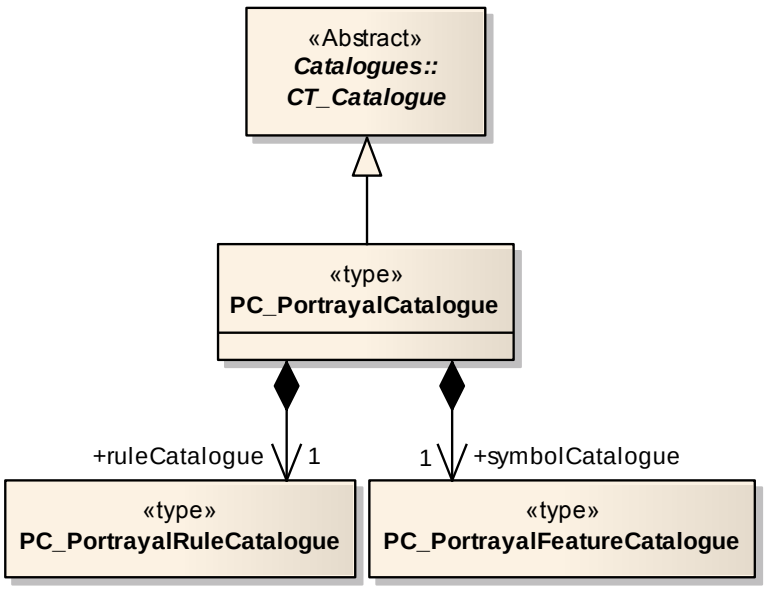

Figure 48 — PC_PortrayalCatalogue context diagram

10.2.8.2 Generalization – CT_Catalogue

PC_PortrayalCatalogue specializes the abstract type "CT_Catalogue" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.2.8.3 Aggregation role – ruleCatalogue

The composition role "ruleCatalogue" specifies the portrayal rule catalogue component of a portrayal catalogue.

```
PC_PortrayalCatalogue::ruleCatalogue[1] : PC_PortrayalRuleCatalogue
```

10.2.8.4 Aggregation role – symbolCatalogue

The composition role "symbolCatalogue" specifies the portrayal feature catalogue component of a portrayal catalogue.

```
PC_PortrayalCatalogue::symbolCatalogue[1] :
     PC_PortrayalFeatureCatalogue
```

10.2.9 Type – PC_PortrayalRuleCatalogue

10.2.9.1 Class semantics

PC_PortrayalRuleCatalogue specializes CT_Catalogue containing a number of portrayal mappings with their component rule statements.

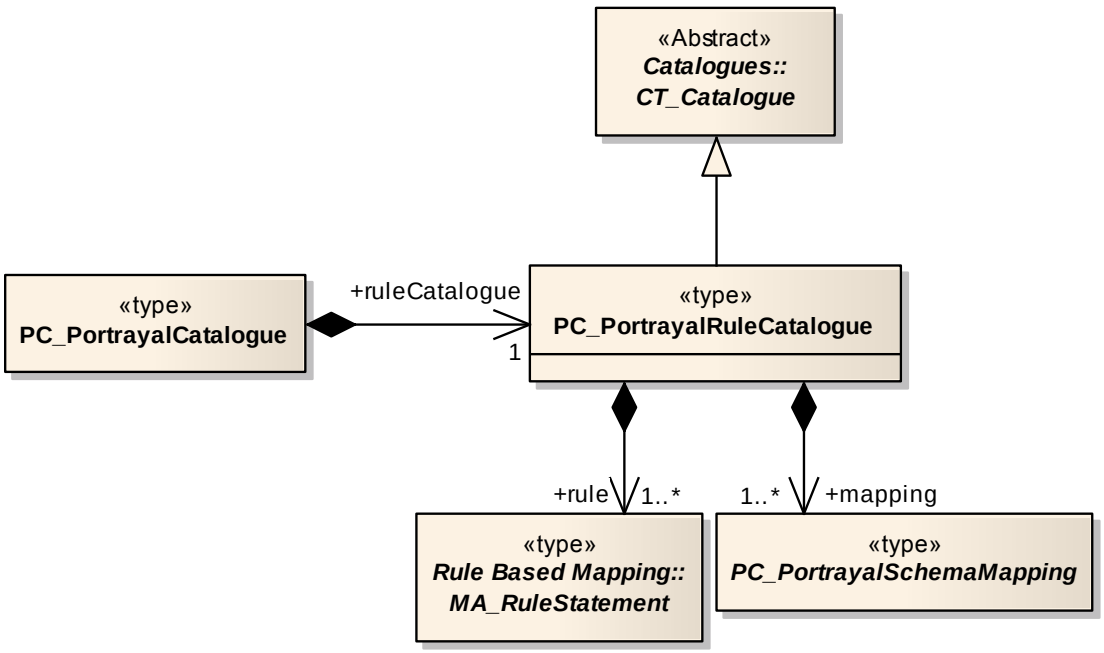

Figure 49 — PC_PortrayalRuleCatalogue context diagram

10.2.9.2 Generalization – CT_Catalogue

PC_PortrayalRuleCatalogue specializes the abstract type "CT_Catalogue" for portrayal rules. As such it shall implement all inherited attributes, operations and associations.

10.2.9.3 Aggregation role – rule

The composition role "rule" specifies the rule statements that comprise a portrayal rule catalogue.

```
PC_PortrayalRuleCatalogue::rule[1..*] : MA_RuleStatement
```

10.2.9.4 Aggregation role – mapping

The composition role "mapping" specifies the portrayal feature mappings that comprise a portrayal rule catalogue.

```
PC_PortrayalRuleCatalogue::mapping[1..*] : PC_PortrayalSchemaMapping
```

10.3 Package – Portrayal Instance

10.3.1 Package semantics

The Portrayal Instance package specializes instances of portrayal features from the general feature model. The pack contains 6 classes: one metaclass, one generic portrayal feature type, and four types. The metaclass GP_PortrayalFeatureType specializes the GFM for portrayal. The portrayal feature type PF_PortrayalFeature is an abstract root class for portrayal features. The two types PF_PortrayalFeatureTypeAttribute and PF_PortrayalSubFeature specialize instances of portrayal feature properties. The two types PF_DisplayParameter and PF_DisplayPriority define display specific elements of a portrayal.

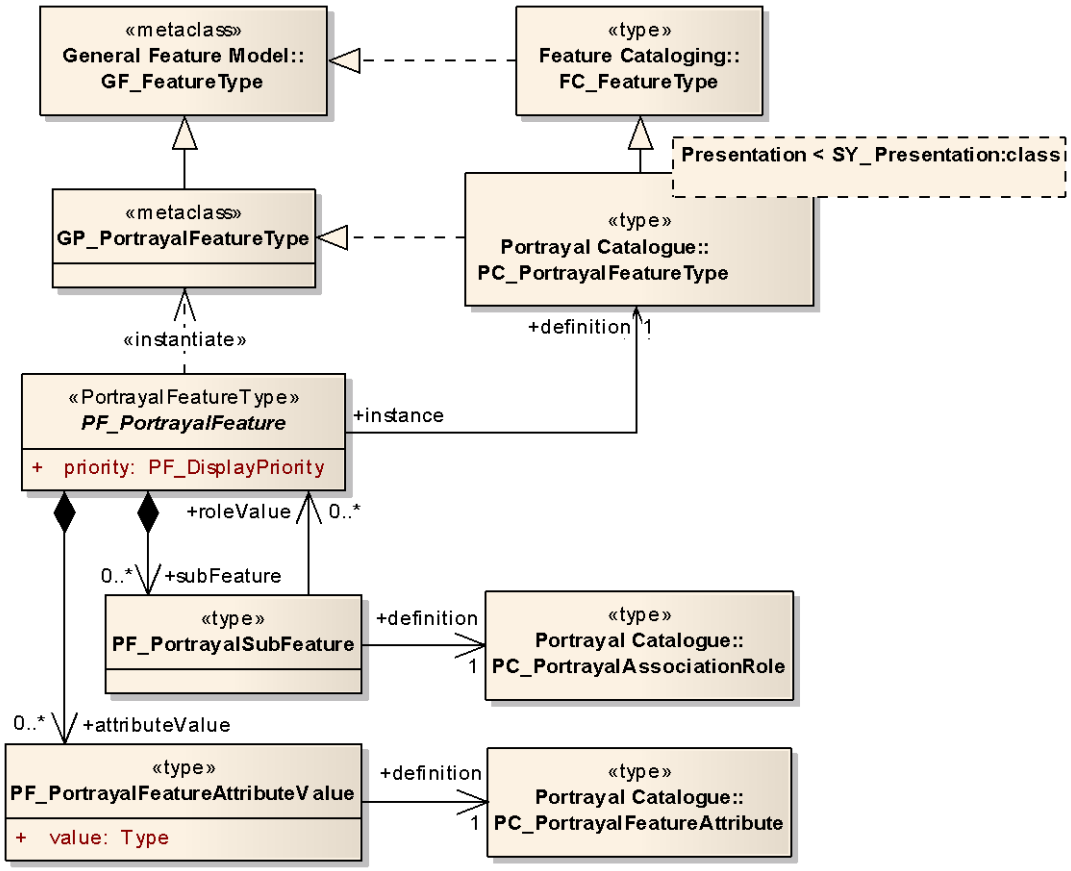

Figure 50 — Portrayal Feature Instance

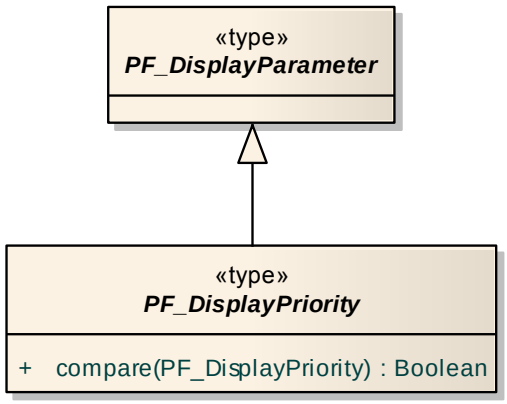

Figure 51 — Display Parameter

10.3.2 Metaclass – GP_PortrayalFeatureType

10.3.2.1 Class semantics

GP_PortrayalFeatureType specializes the metaclass "GF_FeatureType" and is an abstraction of real world phenomena for portrayal.

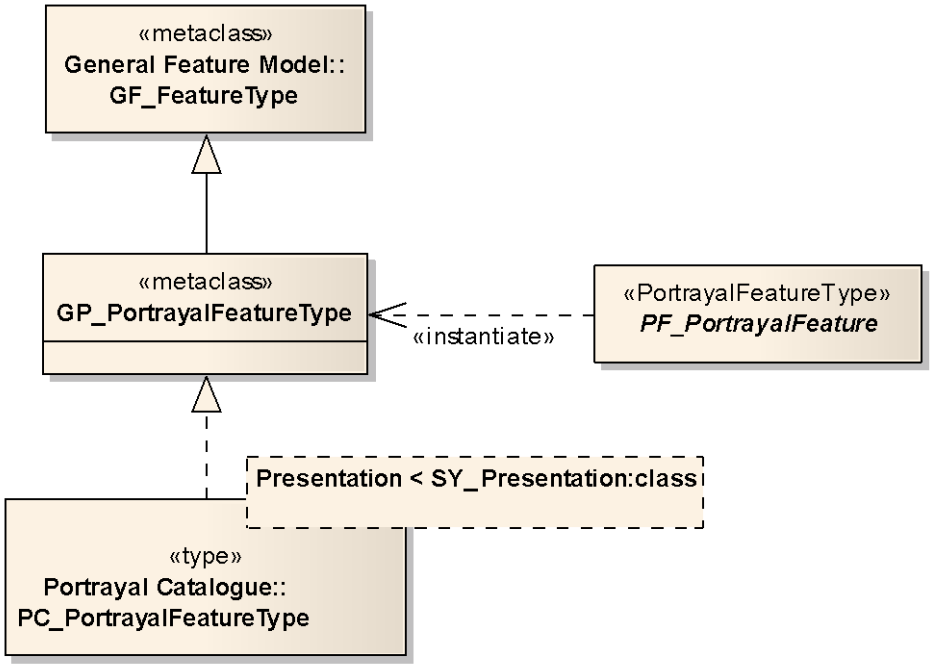

Figure 52 — GP_PortrayalFeatureType context diagram

10.3.2.2 Generalization – GF_FeatureType

GP_PortrayalFeatureType specializes the metaclass "GF_FeatureType" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.3.3 Portrayal Feature Type – PF_PortrayalFeature

10.3.3.1 Class semantics

PF_PortrayalFeature is the basic portrayal feature data structure instantiating GP_PortrayalFeatureType. The composition of a PF_PortrayalFeature is specified by a PC_PortrayalFeatureType.

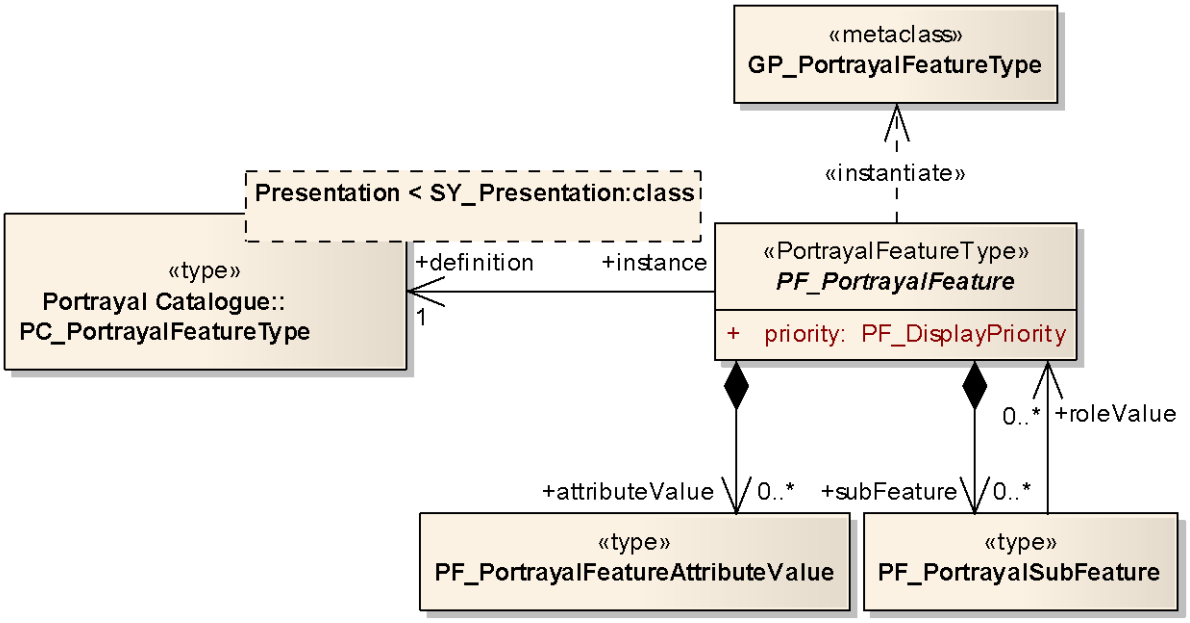

Figure 53 — PF_PortrayalFeature context diagram

10.3.3.2 Association role – definition

The association role "definition" relates an instance of a portrayal feature with its portrayal feature type definition.

```
PF_PortrayalFeature::definition[1] : PC_PortrayalFeatureType
```

10.3.3.3 Aggregation role – attributeValue

The composition role "attributeValue" specifies in a generic manner the attribute values of a portrayal feature.

```
PF_PortrayalFeature::attributeValue[0..*] :
     PF_PortrayalFeatureAttributeValue
```

10.3.3.4 Aggregation role – subFeature

The composition role "subFeature" specifies in a generic manner the subfeature relationships of a portrayal feature.

```
PF_PortrayalFeature::subFeature[0..*] : PF_PortrayalSubFeature
```

10.3.3.5 Attribute – priority

The attribute "priority" specifies the priority of the presentation of a portrayal feature.

```
PF_PortrayalFeature::priority : PF_DisplayPriority
```

10.3.4 Type – PF_PortrayalFeatureAttributeValue

10.3.4.1 Class semantics

PF_PortrayalFeatureAttributeValue is a generic specification of a portrayal feature attribute value.

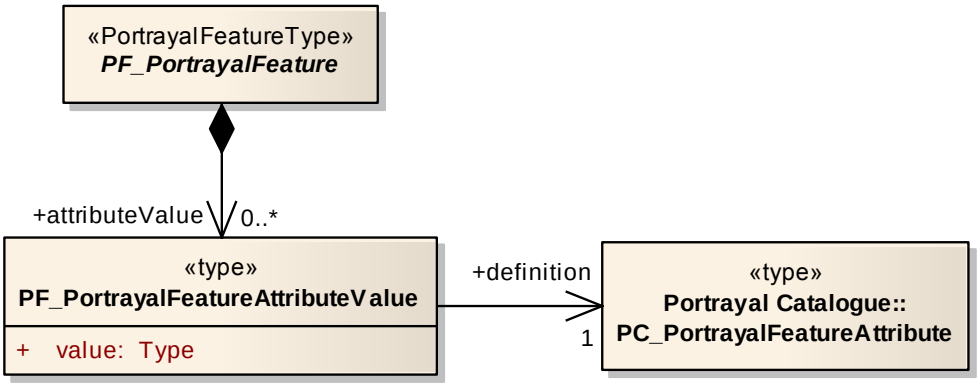

Figure 54 — PF_PortrayalFeatureAttributeValue context diagram

10.3.4.2 Association role – definition

The association role "definition" relates an instance of a portrayal feature attribute value with its portrayal feature attribute definition.

```
PF_PortrayalFeatureAttributeValue::definition[1] :
     PC_PortrayalFeatureAttribute
```

10.3.4.3 Attribute – value

The attribute "value" is a generic definition of the value contained by a portrayal feature attribute value.

```
PF_PortrayalFeatureAttributeValue::value : Type
```

10.3.5 Type – PF_PortrayalSubFeature

10.3.5.1 Class semantics

PF_PortrayalSubFeature is a generic specification of a portrayal subfeature association role.

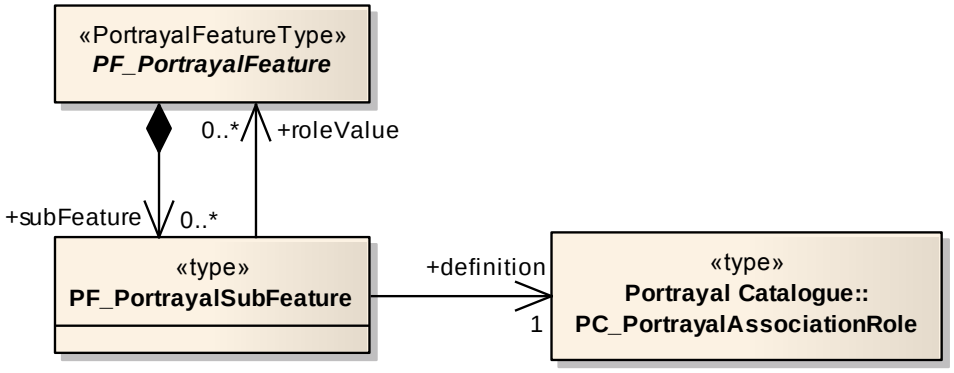

Figure 55 — PF_PortrayalSubFeature context diagram

10.3.5.2 Association role – roleValue

The association role "roleValue" relates in a generic manner an instance of a portrayal subfeature association role with the associated subfeatures.

```
PF_PortrayalSubFeature::roleValue[0..*] : PF_PortrayalFeature
```

10.3.5.3 Association role – definition

The association role "definition" relates an instance of a portrayal subfeature association role with its portrayal association role definition.

```
PF_PortrayalSubFeature::definition[1] : PC_PortrayalAssociationRole
```

10.3.6 Type – PF_DisplayParameter

10.3.6.1 Class semantics

PF_DisplayParameter is the abstract root type describing display specific parameters of a portrayal feature. This can be specialized to handle display specific aspects such as display priority, or S-57 viewing groups or radar flags.

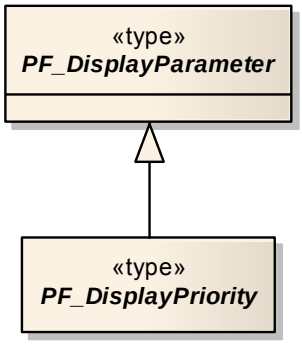

Figure 56 — PF_DisplayParameter context diagram

10.3.6.2 Generalization – PF_DisplayParameter

PF_DisplayPriority specializes the abstract type "PF_DisplayParameter". As such it shall implement all inherited attributes, operations and associations.

10.3.7 Type – PF_DisplayPriority

10.3.7.1 Class semantics

PF_DisplayPriority is an abstract type specifying display priority. This is the used to determine the order in which symbols and symbol elements are displayed. A realization of this type will allow a total ordering of portrayal features for display.

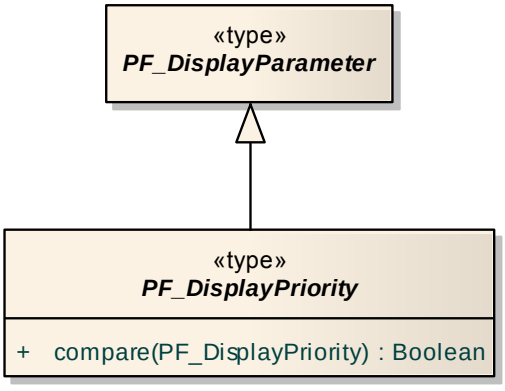

Figure 57 — PF_DisplayPriority context diagram

10.3.7.2 Generalization – PF_DisplayParameter

PF_DisplayPriority specializes the abstract type "PF_DisplayParameter". As such it shall implement all inherited attributes, operations and associations.

10.3.7.3 Operation – compare

The operation "compare" shall accept a display priority comparing it with the current display priority and returning a Boolean value that allows a total ordering.

```
PF_DisplayPriority::compare(
     priority : PF_DisplayPriority
  ) : Boolean
```

10.4 Package – Portrayal Specification

10.4.1 Package semantics

The Portrayal Specification package specializes the feature specification aspect of the Rule Based Mapping. It specializes MA_FeatureSpecification and associated types. The package contains 6 types: PF_PortrayalSpecification, PF_InlinePortrayal, PF_PortrayalFeatureConstructor, PF_PortrayalPropertyAssignment, PF_PortrayalAttributeAssignment, and PF_PortrayalSubFeatureAssignment. All these types specialize the feature specification for portrayal.

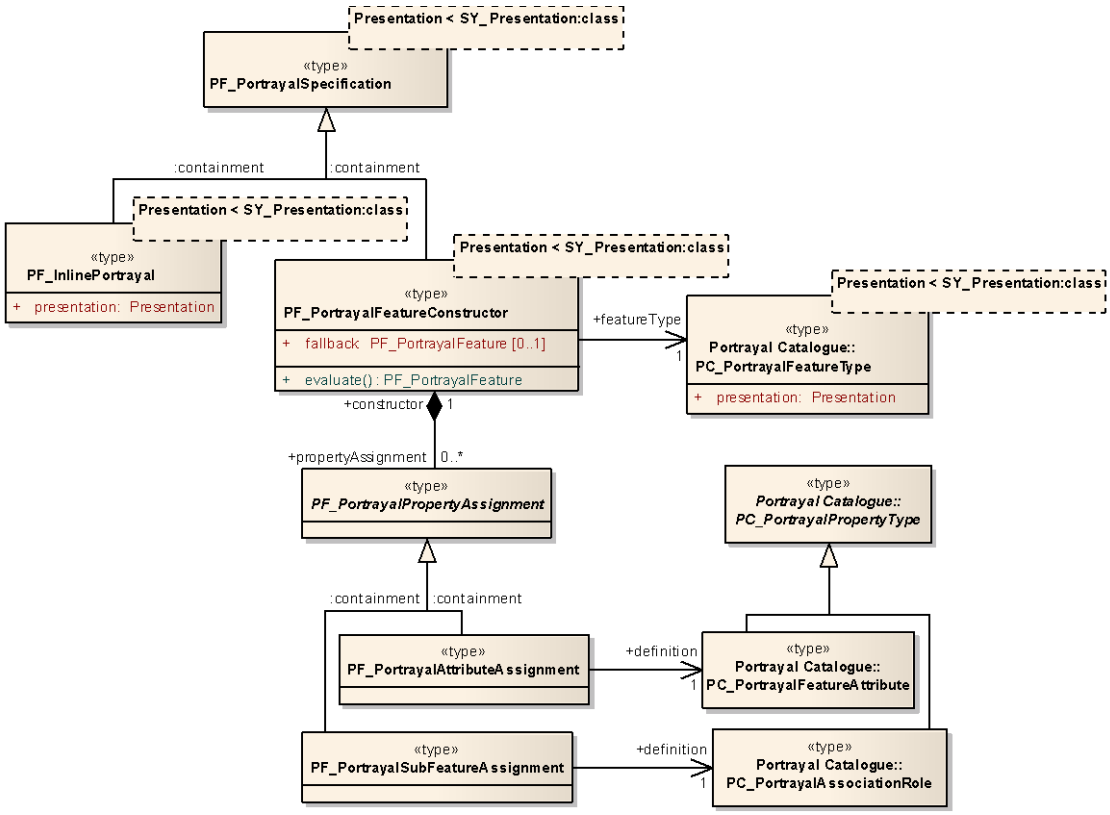

Figure 58 — Portrayal Specification

10.4.2 Type – PF_PortrayalSpecification

10.4.2.1 Class semantics

PF_PortrayalSpecification is a specialization for portrayal of the type "MA_FeatureSpecification" templatized on a specialization of SY_Presentation.

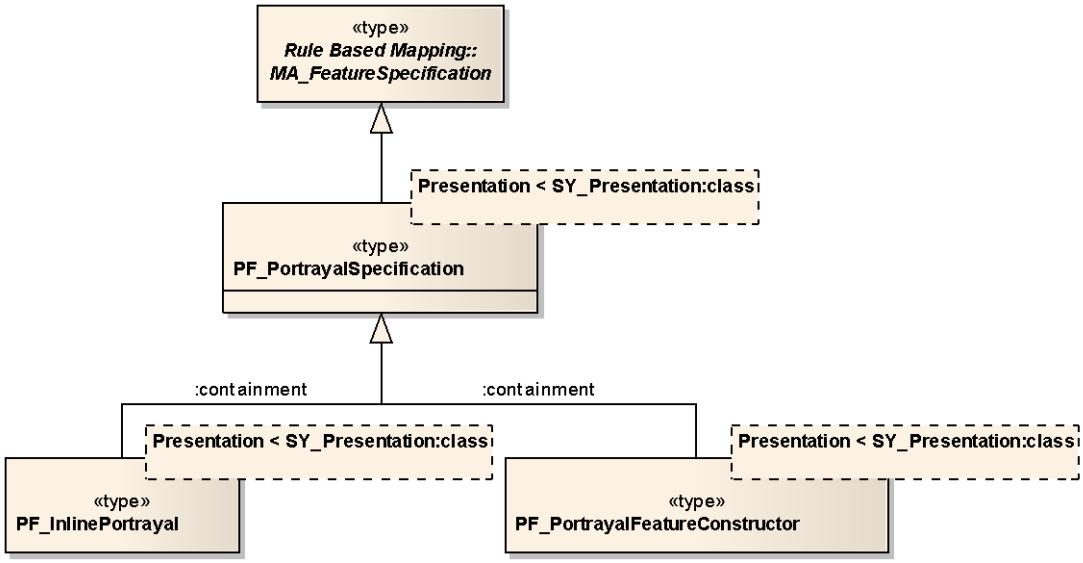

Figure 59 — PF_PortrayalSpecification context diagram

10.4.2.2 Generalization – MA_FeatureSpecification

PF_PortrayalSpecification specializes the abstract type "MA_FeatureSpecification" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.4.3 Type – PF_InlinePortrayal

10.4.3.1 Class semantics

PF_InlinePortrayal specializes PF_PortrayalSpecification, as MA_InlineFeature specializes MA_FeatureSpecification. Instead of containing a whole portrayal feature, PF_InlinePortrayal only contains a presentation. PF_InlinePortrayal also specializes MA_InlineFeature for portrayal.

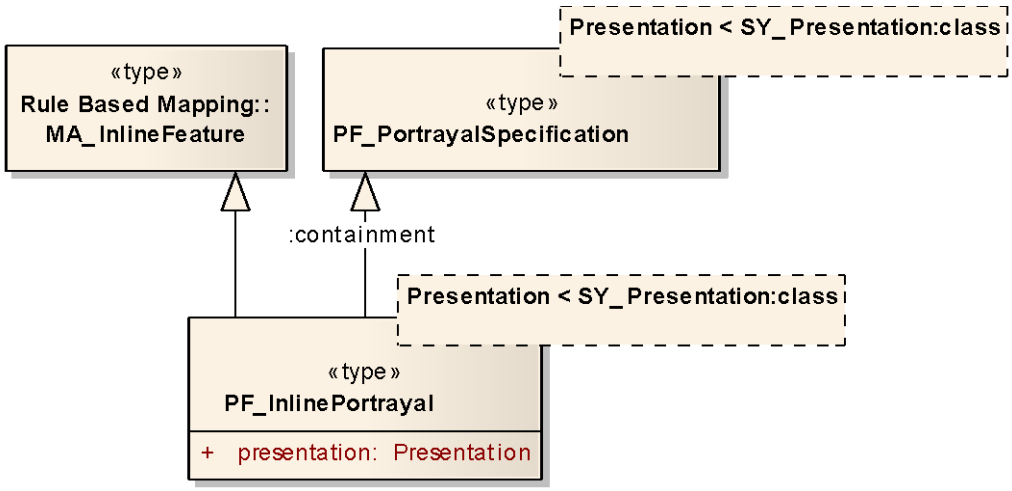

Figure 60 — PF_InlinePortrayal context diagram

10.4.3.2 Generalization – PF_PortrayalSpecification

PF_InlinePortrayal specializes the type "PF_PortrayalSpecification". As such it shall implement all inherited attributes, operations and associations.

10.4.3.3 Generalization MA_InlineFeature

PF_InlinePortrayal specializes the type "MA_InlineFeature" for portrayal. As such it shall implement all inherited attributes, operations and associations, where as all FD_Feature references are replace with the "Presentation" template parameter.

10.4.3.4 Attribute – presentation

The attribute "presentation" specifies the presentation or symbol of the inline portrayal specification. The template type specifies the type of the presentation and is a specialization of SY_Presentation.

```
PF_InlinePortrayal::presentation : Presentation
```

10.4.4 Type – PF_PortrayalFeatureConstructor

10.4.4.1 Class semantics

PF_PortrayalFeatureConstructor specializes PF_PortrayalSpecification, as MA_FeatureConstructor specializes MA_FeatureSpecification. PF_PortrayalFeatureConstructor also specializes MA_FeatureConstructor for portrayal, specifying how to create a portrayal feature from input data.

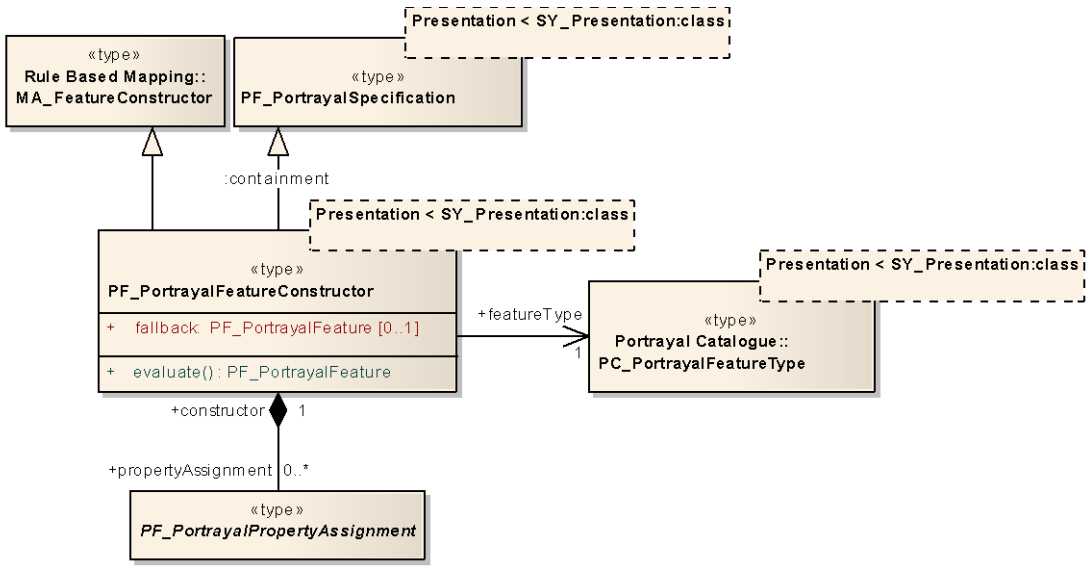

Figure 61 — PF_PortrayalFeatureConstructor context diagram

10.4.4.2 Generalization – PF_PortrayalSpecification

PF_PortrayalFeatureConstructor specializes the type "PF_PortrayalSpecification". As such it shall implement all inherited attributes, operations and associations.

10.4.4.3 Generalization – MA_FeatureConstructor

PF_PortrayalFeatureConstructor the type "MA_FeatureConstructor" for portrayal. As such it shall implement all inherited attributes, operations and associations, where as all FD_Feature references are replace with PF_PortrayalFeature.

10.4.4.4 Association role – featureType

The association role "featureType" specifies the portrayal feature type of the portrayal feature that is created.

```
PF_PortrayalFeatureConstructor::featureType[1] :
     PC_PortrayalFeatureType
```

10.4.4.5 Aggregation role – propertyAssignment

The composition role "propertyAssignment" specifies the property assignments used to populate a newly created portrayal feature.

```
PF_PortrayalFeatureConstructor::propertyAssignment[0..*] :
     PF_PortrayalPropertyAssignment
```

10.4.4.6 Attribute – fallback

The optional attribute "fallback" is a portrayal feature instance returned by the "evaluate" operation should construction of the portrayal feature fail.

```
PF_PortrayalFeatureConstructor::fallback : PF_PortrayalFeature  [0..1]
```

10.4.4.7 Operation – evaluate

The operation "evaluate" shall accept a feature and a context record as input and return a portrayal feature as the result of the constructor.

```
PF_PortrayalFeatureConstructor::evaluate(
     feature : FD_Feature[1..*]
     context : Record
   ) : PF_PortrayalFeature
```

10.4.5 Type – PF_PortrayalPropertyAssignment

10.4.5.1 Class semantics

PF_PortrayalPropertyAssignment is an abstract property assignment for specifying a value for a portrayal feature property as part of a portrayal feature constructor.

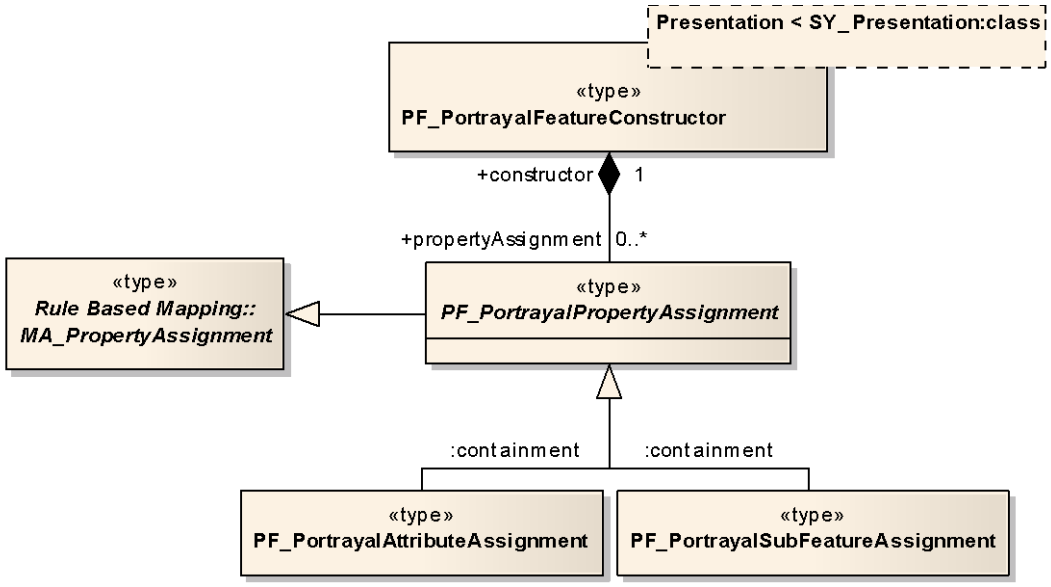

Figure 62 — PF_PortrayalPropertyAssignment context diagram

10.4.5.2 Generalization – MA_PropertyAssignment

PF_PortrayalPropertyAssignment specializes the abstract type "MA_PropertyAssignment" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.4.5.3 Aggregation role – constructor

The composition role "constructor" references the containing portrayal feature constructor for the property assignment.

```
PF_PortrayalPropertyAssignment::constructor[1] :
     PF_PortrayalFeatureConstructor
```

10.4.6 Type – PF_PortrayalAttributeAssignment

10.4.6.1 Class semantics

PF_PortrayalAttributeAssignment specializes PF_PortrayalPropertyAssignment. It assigns a value to an attribute of a newly created portrayal feature from a value expression. PF_PortrayalAttributeAssignment also specializes MA_AttributeAssignment for portrayal.

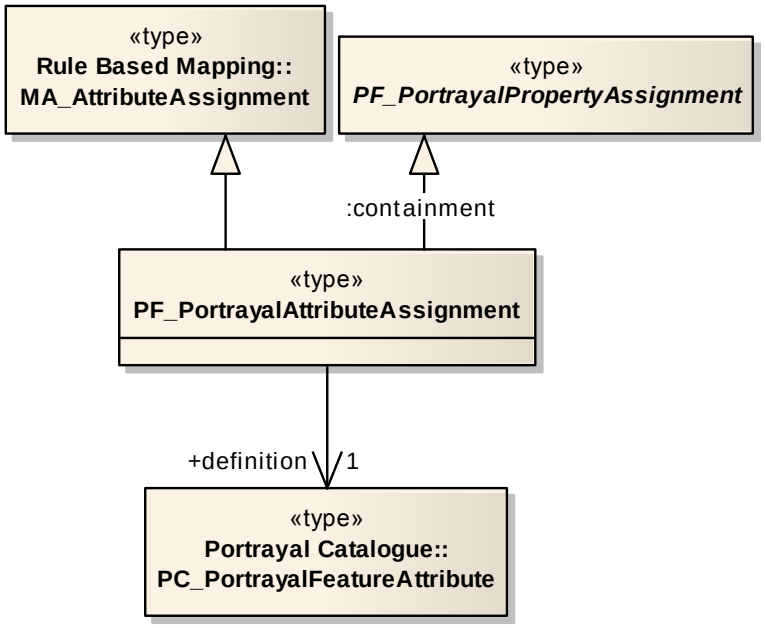

Figure 63 — PF_PortrayalAttributeAssignment context diagram

10.4.6.2 Generalization – PF_PortrayalPropertyAssignment

PF_PortrayalAttributeAssignment specializes the abstract type "PF_PortrayalPropertyAssignment" as part of the "containment" generalization set. As such it shall implement all inherited attributes, operations and associations.

10.4.6.3 Generalization – MA_AttributeAssignment

PF_PortrayalAttributeAssignment specializes the abstract type "MA_AttributeAssignment" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.4.6.4 Association role – definition

The association role "definition" specifies the portrayal feature attribute to which a value is assigned in the newly created portrayal feature.

```
PF_PortrayalAttributeAssignment::definition[1] :
     PC_PortrayalFeatureAttribute
```

10.4.7 Type – PF_PortrayalSubFeatureAssignment

10.4.7.1 Class semantics

PF_PortrayalSubFeatureAssignment specializes PF_PortrayalPropertyAssignment. It creates a relationship between a newly created portrayal feature and an existing portrayal feature. PF_PortrayalSubFeatureAssignment also specializes MA_AssociationAssignment for portrayal.

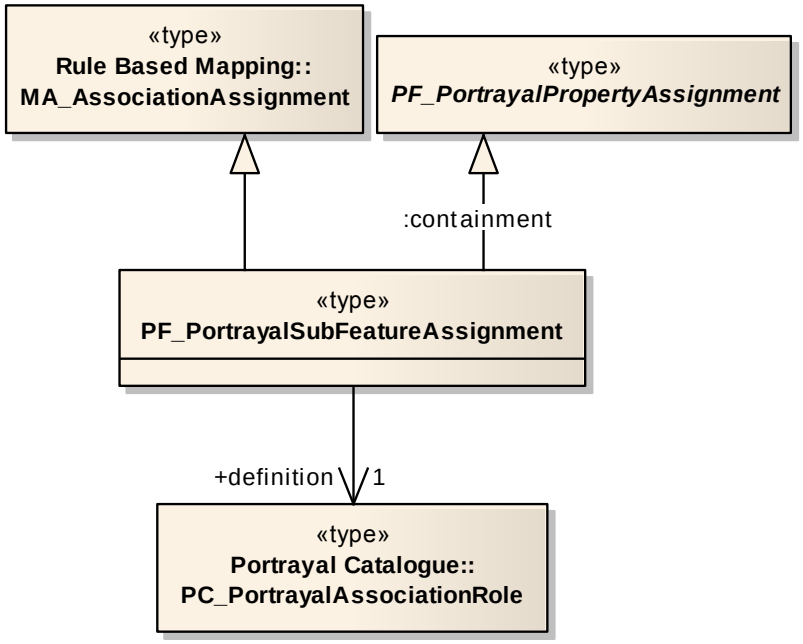

Figure 64 — PF_PortrayalSubFeatureAssignment context diagram

10.4.7.2 Generalization – PF_PortrayalPropertyAssignment

PF_PortrayalSubFeatureAssignment specializes the abstract type "PF_PortrayalPropertyAssignment" as part of the "containment" generalization set. As such it shall implement all inherited attributes, operations and associations.

10.4.7.3 Generalization – MA_AssociationAssignment

PF_PortrayalSubFeatureAssignment specializes the type "MA_AssociationAssignment" for portrayal. As such it shall implement all inherited attributes, operations and associations.

10.4.7.4 Association role – definition

The association role "definition" specifies the association role to which a portrayal feature is related in the newly created portrayal feature.

```
PF_PortrayalSubFeatureAssignment::definition[1] :
     PC_PortrayalAssociationRole
```

11 Package – Presentation

11.1 Package semantics

There are various ways to portray geographic information. Most prevalent are various types of visual, delineation dependent, graphic symbols, although tactile, aural, and others methods of presentation are also possible. This package defines a root for various forms of presentation and defines subpackages for each separate one. This package has a root subpackage and four delineation dependent subpackages: Point, Line, Area, and Text.

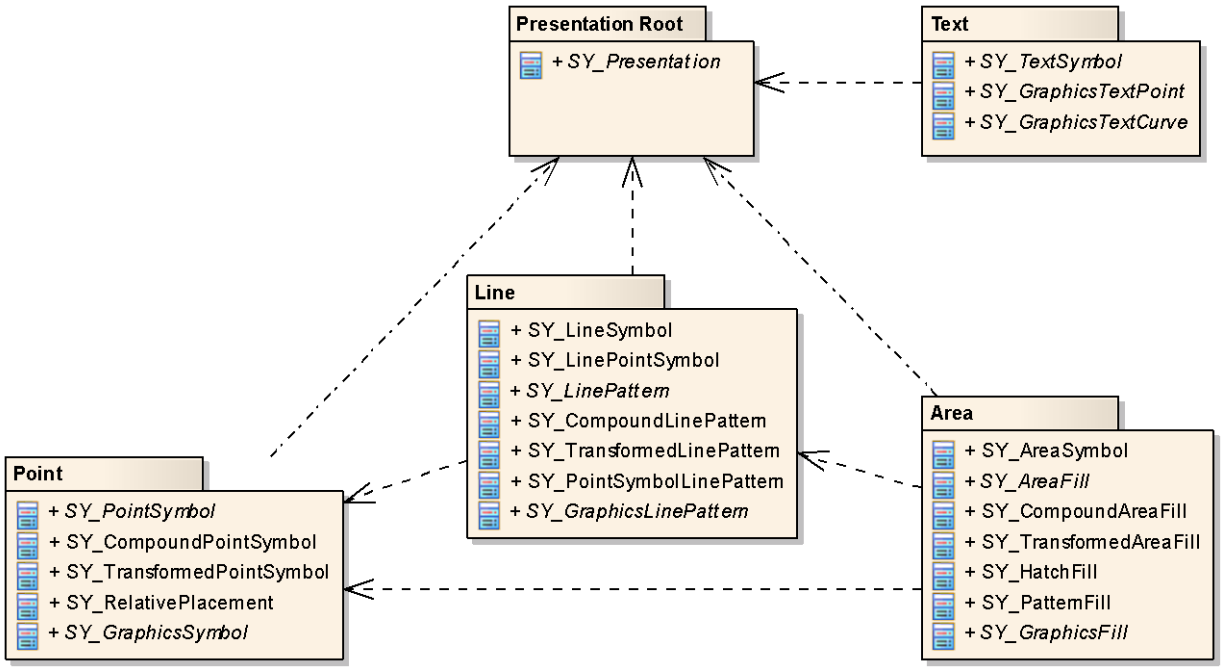

Figure 65 — Package Overview

Symbols are initially without meaning and without location. When they become part of a portrayal they gain both meaning and location both with respect to the portrayal and with respect to the universe they represent. The definition of a symbol has its own engineering coordinate reference system. A coordinate reference system transformation places a point symbol into a portrayal (Figure 66). Similarly a coordinate reference system transformation places a subsymbol into a parent point symbol.

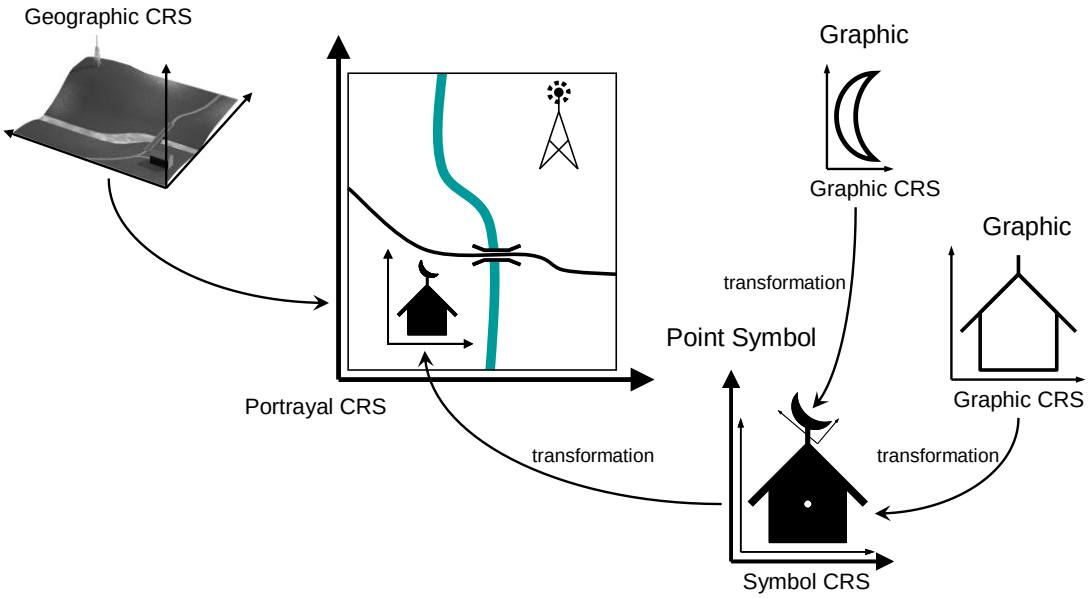

Figure 66 — Portrayal and Symbol CRSs

Line symbols and patterns have two kinds of coordinate reference systems. A line coordinate reference system is used to define the pen or line style. This coordinate reference system is perpendicular to the geometry of the curve and allows for the specification of line widths and offsets. The second kind of coordinate reference system is a local coordinate reference system which is define for every location along a curve. This coordinate reference system has an x-axis that is tangential to the curve and a y-axis perpendicular to the x-axis (Figure 67).

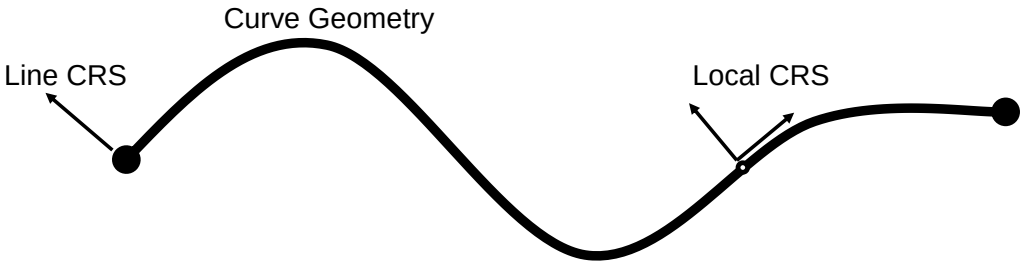

Figure 67 — Line and Local CRSs

An area symbol defines coordinate reference systems for its boundary and for its interior. The boundary coordinate reference systems are those defined for line symbols. The interior of the area symbol has its own coordinate reference systems (Figure 68).

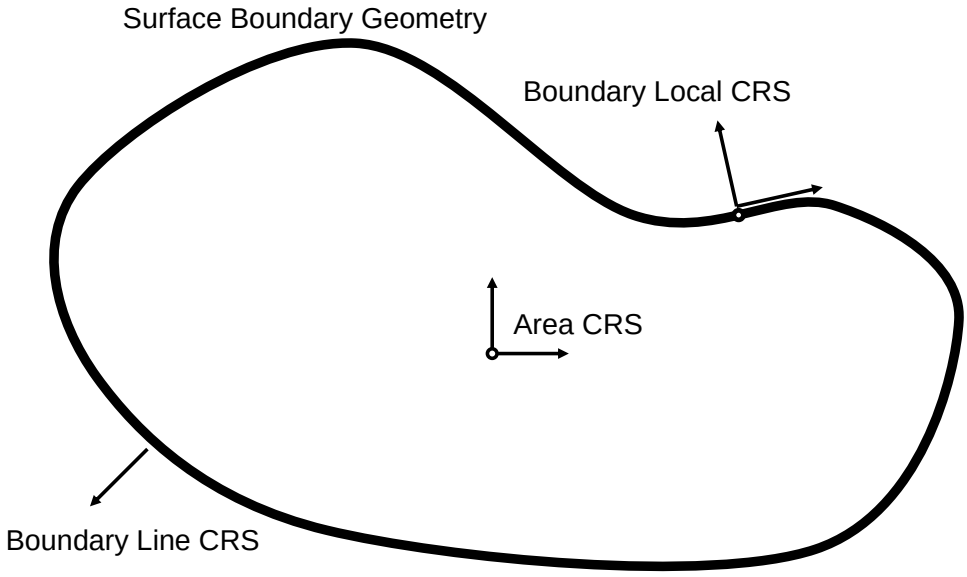

Figure 68 — Area and Boundary CRSs

Should the area symbol include a tiled pattern area fill then there is a tile coordinate reference system as well (Figure 69).

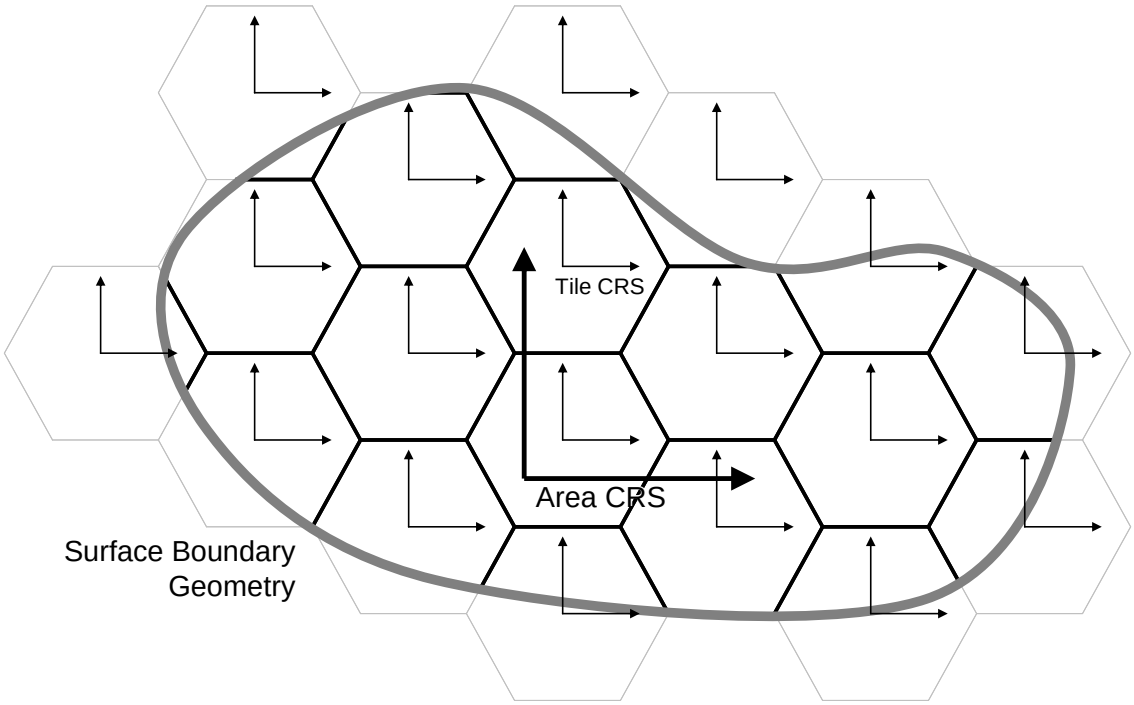

Figure 69 — Tile CRS

11.2 Package – Presentation Root

11.2.1 Package semantics

The Presentation Root package is the root package for presentation. It contains the type SY_Presentation, which is abstract root type for presentation. This root type is specialized by the types: SY_PointSymbol, SY_LineSymbol, SY_AreaSymbol, SY_TextSymbol, SY_LinePattern, and SY_AreaFill.

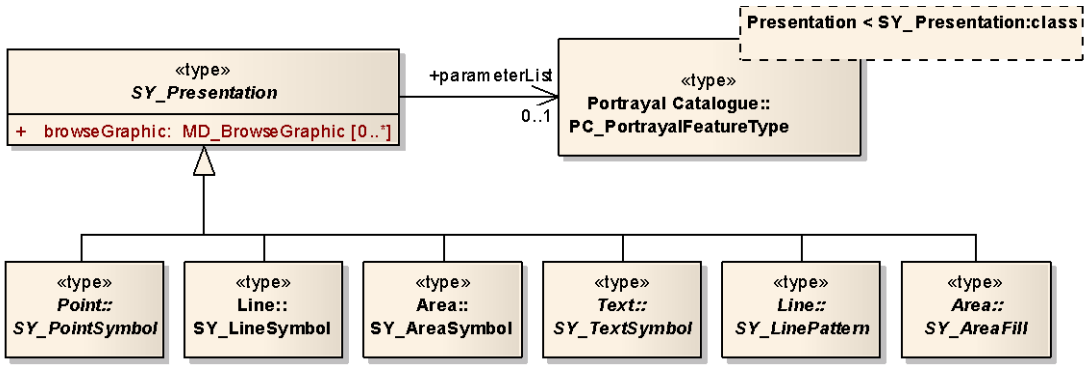

Figure 70 — Presentation

11.2.2 Type – SY_Presentation

11.2.2.1 Class semantics

SY_Presentation is the abstract root type for presentation. It defines a generic presentation with browse graphics and a portrayal feature type for the interface with parameters, where the parameters to the presentation are the attributes of the portrayal feature type.

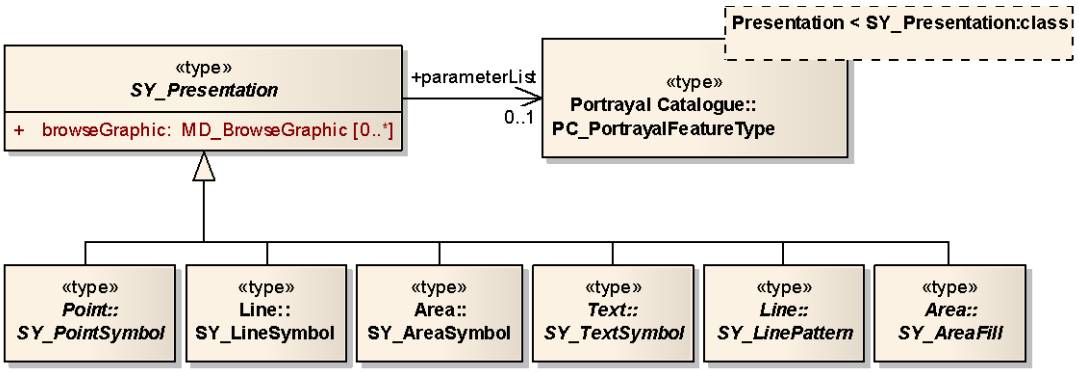

Figure 71 — SY_Presentation context diagram

11.2.2.2 Association role – parameterList

The optional association role "parameterList" references a portrayal feature type, which defines attributes that function as parameters to the presentation.

```
SY_Presentation::parameterList[0..1] : PC_PortrayalFeatureType
```

11.2.2.3 Attribute – browseGraphic

The optional attribute "browseGraphic" specifies graphics that may be used as metadata for the presentation and used to give a sample of the appearance of the presentation.

```
SY_Presentation::browseGraphic : MD_BrowseGraphic  [0..*]
```

11.3 Package – Point

11.3.1 Package semantics

The Point package specifies point symbols. A point symbol is a symbol attached to a location. A point symbol is composed of graphics and other point symbol placed with a transformation. The graphics of the point symbol use existing graphic standards.

The package contains 4 types and a code list. The type SY_PointSymbol is the abstract root specialized by SY_GraphicsSymbol, SY_TransformedPointSymbol, and SY_CompoundPointSymbol. A SY_GraphicsSymbol is the graphic basis of a point symbol. SY_CompoundPointSymbol composes multiple point symbols into one, and SY_TransformedPointSymbol transforms a point symbol. The code list SY_RelativePlacement specifies if a transformation is relative to the superordinate point symbol or the portrayal.

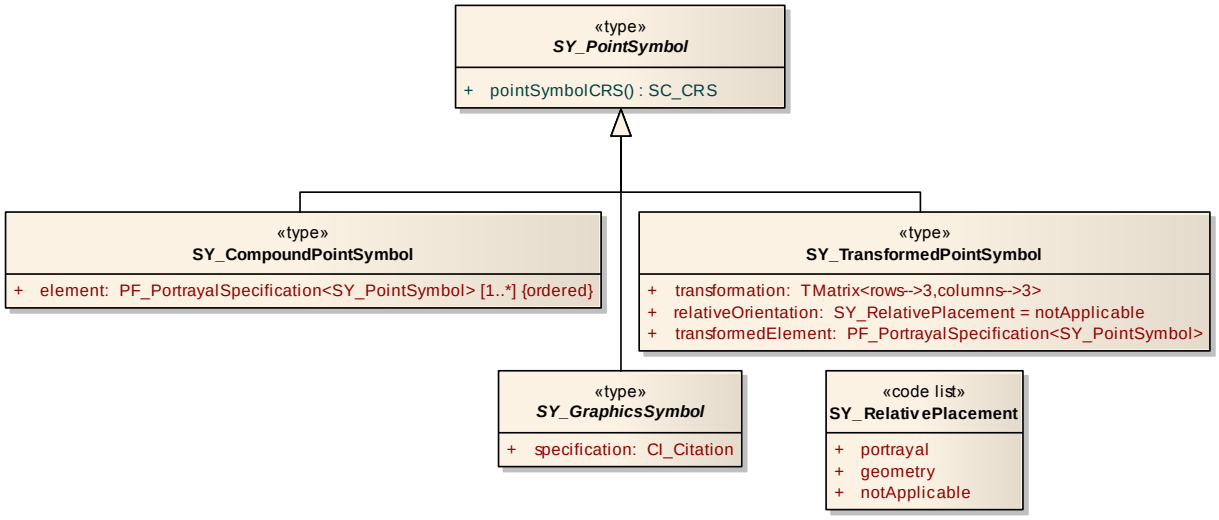

Figure 72 — Point Symbol

11.3.2 Type – SY_PointSymbol

11.3.2.1 Class semantics

SY_PointSymbol specializes SY_Presentation as abstract type for point symbol presentation.

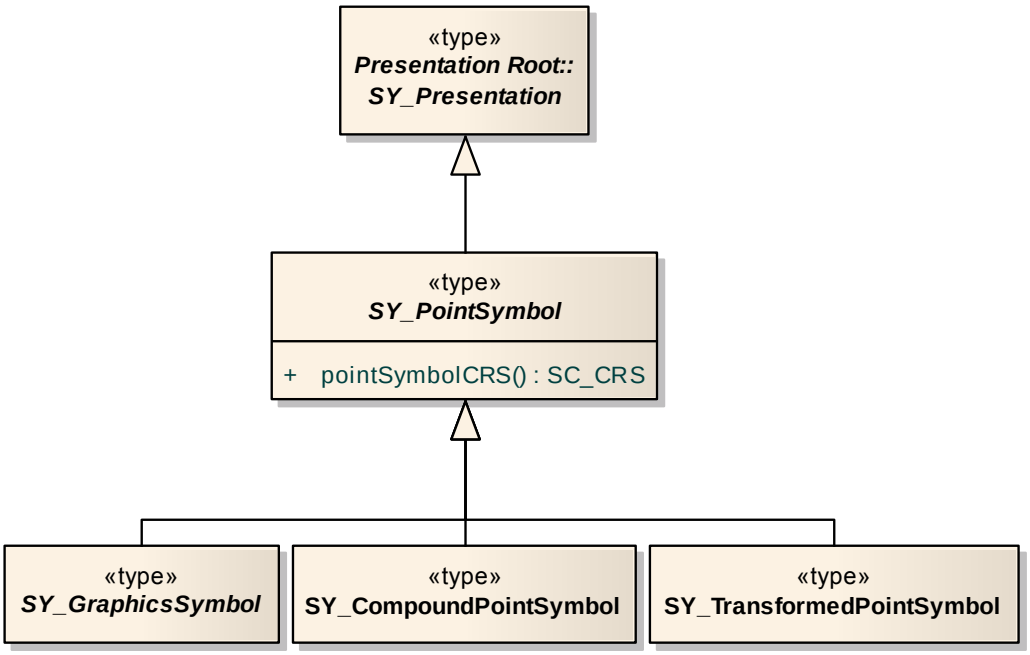

Figure 73 — SY_PointSymbol context diagram

11.3.2.2 Generalization – SY_Presentation

SY_PointSymbol specializes the abstract root type "SY_Presentation" as a point symbol. As such it shall implement all inherited attributes, operations and associations.

11.3.2.3 Operation – pointSymbolCRS

The operation "pointSymbolCRS" returns the coordinate reference system of the point symbol presentation.

```
SY_PointSymbol::pointSymbolCRS(
  ) : SC_CRS
```

11.3.3 Type – SY_CompoundPointSymbol

11.3.3.1 Class semantics

SY_CompoundPointSymbol specializes SY_PointSymbol as a composition of subsymbols. The subsymbols may either be inline or referenced as portrayal features. The elements of the composition are ordered for display.

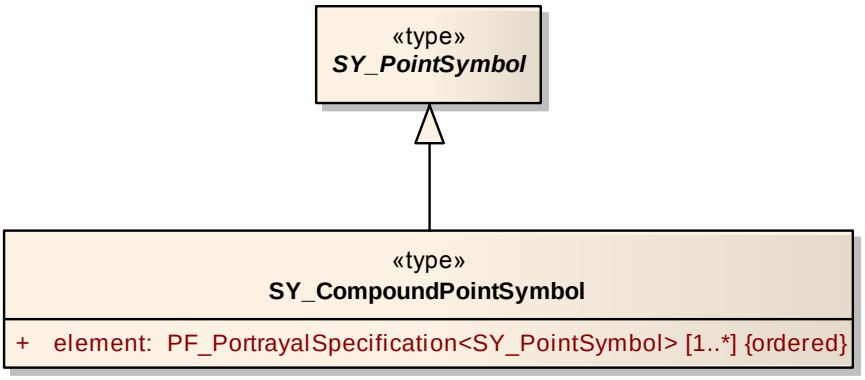

Figure 74 — SY_CompoundPointSymbol context diagram

11.3.3.2 Generalization – SY_PointSymbol

SY_CompoundPointSymbol specializes the type "SY_PointSymbol" as a compound of multiple subsymbol elements. As such it shall implement all inherited attributes, operations and associations.

11.3.3.3 Attribute – element

The attribute "element" specifies the ordered list of subsymbol elements of a compound point symbol.

```
SY_CompoundPointSymbol::element :
     PF_PortrayalSpecification<SY_PointSymbol>  [1..*]
```

11.3.4 Type – SY_TransformedPointSymbol

11.3.4.1 Class semantics

SY_TransformedPointSymbol specializes SY_PointSymbol as a geometric transformation of the original symbol. The transformed symbol may either be inline or referenced as a portrayal feature.

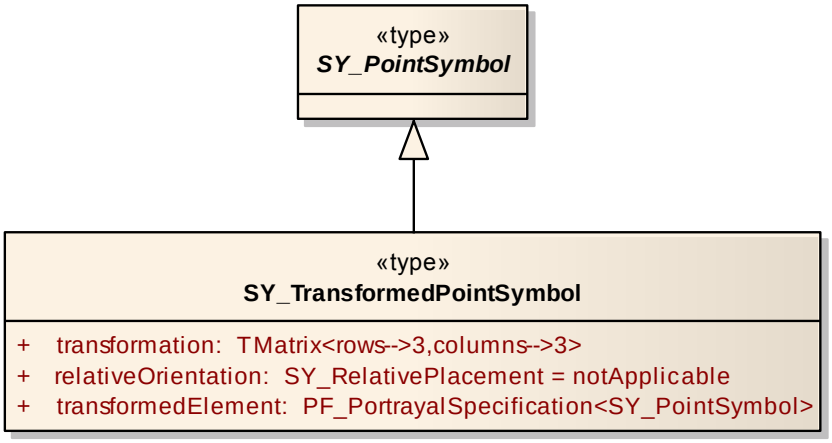

Figure 75 — SY_TransformedPointSymbol context diagram

11.3.4.2 Generalization – SY_PointSymbol

SY_TransformedPointSymbol specializes the type "SY_PointSymbol" as a geometrically transformed symbol. As such it shall implement all inherited attributes, operations and associations.

11.3.4.3 Attribute – transformation

The attribute "transformation" uses homogeneous coordinates to define a geometric transformation of a point symbol.

```
SY_TransformedPointSymbol::transformation : TMatrix<rows→3,columns→3>
```

11.3.4.4 Attribute – relativeOrientation

The attribute "relativeOrientation" specifies if the transformation is relative to the superordinate presentation, relative to the portrayal coordinate reference system, or not applicable. The default value of this attribute is "notApplicable".

```
SY_TransformedPointSymbol::relativeOrientation : SY_RelativePlacement =
     notApplicable
```

11.3.4.5 Attribute – transformedElement

The attribute "transformedElement" specifies the symbol to be transformed. The transformed symbol may either be inline or referenced as a portrayal feature.

```
SY_TransformedPointSymbol::transformedElement :
     PF_PortrayalSpecification<SY_PointSymbol>
```

11.3.5 Code List – SY_RelativePlacement

11.3.5.1 Class semantics

The code list "SY_RelativePlacement" names the possible relative placements of a presentation, in particular point symbols. A placement can either be relative the coordinate reference system of the portrayal or relative to the coordinate reference system of the superordinate presentation within which the presentation is being placed.

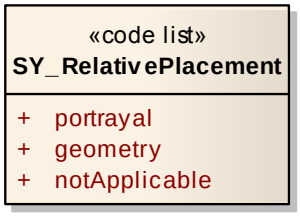

Figure 76 — SY_RelativePlacement context diagram

11.3.5.2 Attribute – portrayal

```
SY_RelativePlacement::portrayal : <undefined>
```

11.3.5.3 Attribute – geometry

```
SY_RelativePlacement::geometry : <undefined>
```

11.3.5.4 Attribute – notApplicable

```
SY_RelativePlacement::notApplicable : <undefined>
```

11.3.6 Type – SY_GraphicsSymbol

11.3.6.1 Class semantics

SY_GraphicsSymbol specializes SY_PointSymbol as an abstract graphic element of a point symbol. This type is realized by a graphic in a defined graphics language.

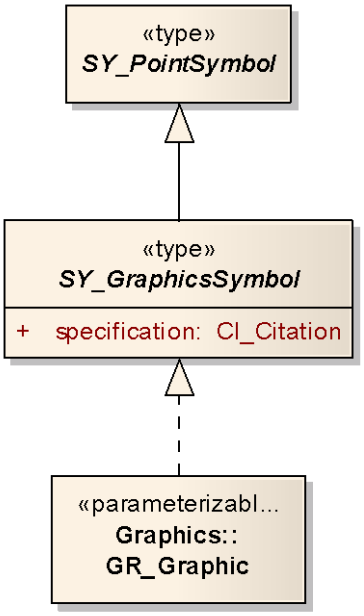

Figure 77 — SY_GraphicsSymbol context diagram

11.3.6.2 Generalization – SY_PointSymbol

SY_GraphicsSymbol specializes the type "SY_PointSymbol" as a graphic representation of the symbol. As such it shall implement all inherited attributes, operations and associations.

11.3.6.3 Attribute – specification

The attribute "specification" cites the specification standard for the graphics definition language used to define the symbol graphic.

```
SY_GraphicsSymbol::specification : CI_Citation
```

11.4 Package – Line

11.4.1 Package semantics

The Line package specifies line symbols and line patterns. A line symbol is composed of a line pattern and possible point symbols along the path of a line. A line pattern is a pattern to render along a path. A line pattern is composed of line graphics and other line patterns placed with a transformation. The graphics of the line pattern use existing graphics standards.

The package contains 7 types. The type SY_LineSymbol specifies a line symbol composed of a line pattern and point symbols, contained in SY_LinePointSymbol, along the path of the line symbol. The type SY_LinePattern is the abstract root specialized by SY_GraphicsLinePattern, SY_PointSymbolLinePattern, SY_TransformedLinePattern, and SY_CompoundLinePattern. A SY_GraphicsLinePattern is the graphic basis of a line pattern. SY_PointSymbolLinePattern is a pattern of repeated point symbols along the path of the line pattern. SY_CompoundLinePattern composes multiple line patterns into one, and SY_TransformedLinePattern transforms a line pattern.

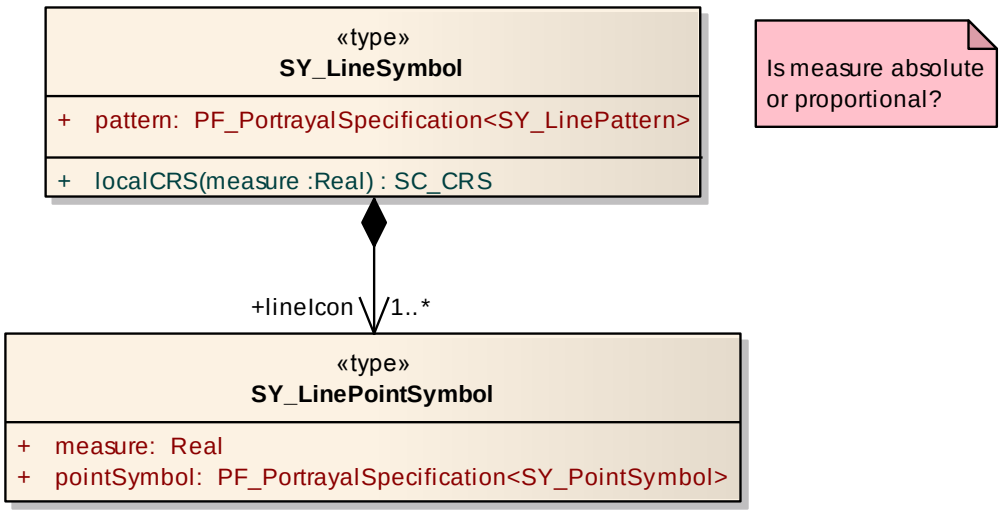

Figure 78 — Line Symbol

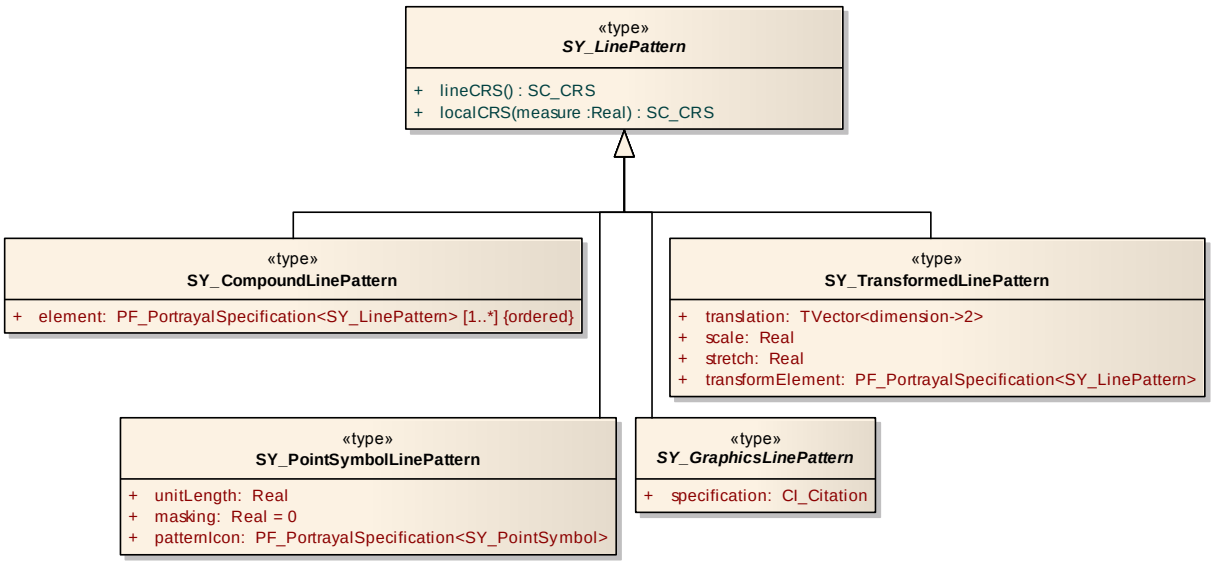

Figure 79 — Line Pattern

11.4.2 Type – SY_LineSymbol

11.4.2.1 Class semantics

SY_LineSymbol specializes SY_Presentation as a line symbol that is composed of a line pattern and possible point symbols along the path of a line. The line symbol has a coordinate reference system defined at every point along its path, which has an x-axis that is tangential to the path at that point.

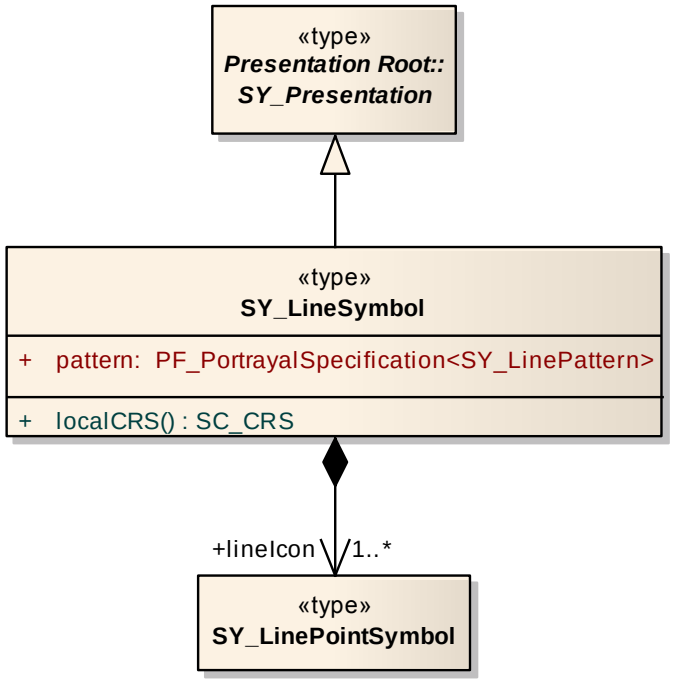

Figure 80 — SY_LineSymbol context diagram

11.4.2.2 Generalization – SY_Presentation

SY_LineSymbol specializes the abstract root type "SY_Presentation" as a line symbol. As such it shall implement all inherited attributes, operations and associations.

11.4.2.3 Aggregation role – lineIcon

The composition role "lineIcon" specifies a point symbol along the path of the line symbol.

```
SY_LineSymbol::lineIcon[1..*] : SY_LinePointSymbol
```

11.4.2.4 Attribute – pattern

The attribute "pattern" specifies the line pattern of the line symbol.

```
SY_LineSymbol::pattern : PF_PortrayalSpecification<SY_LinePattern>
```

11.4.2.5 Operation – localCRS

The operation "localCRS" shall accept a real value as input and return a coordinate reference system. The parameter "measure" specifies the measure along the path of the line symbol specifying the origin of the coordinate reference system. The x-axis is tangential to the path at that point.

```
SY_LineSymbol::localCRS(
     measure : Real
  ) : SC_CRS
```

11.4.3 Type – SY_LinePointSymbol

11.4.3.1 Class semantics

SY_LinePointSymbol specifies a point symbols along the path of the line symbol. The location of the point symbol is specified by a measure along the path. The point symbol may either be inline or referenced as a portrayal feature.

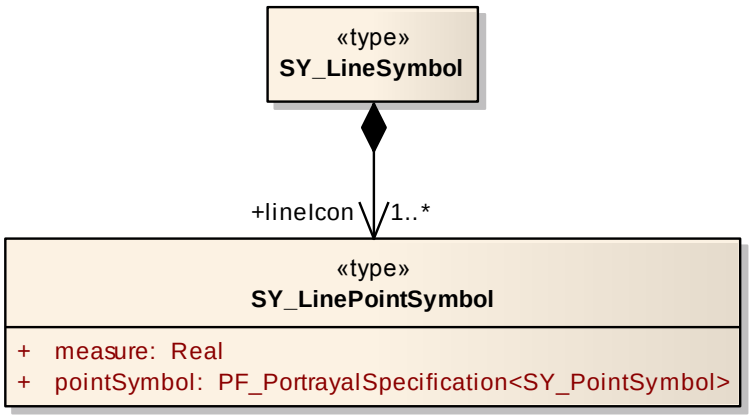

Figure 81 — SY_LinePointSymbol context diagram

11.4.3.2 Attribute – measure

The attribute "measure" specifies the measure along the path for the placement of the point symbol.

```
SY_LinePointSymbol::measure : Real
```

11.4.3.3 Attribute – pointSymbol

The attribute "pointSymbol" specifies the point symbol to be placed along the path of the line symbol. The point symbol may either be inline or referenced as a portrayal feature.

```
SY_LinePointSymbol::pointSymbol :
     PF_PortrayalSpecification<SY_PointSymbol>
```

11.4.4 Type – SY_LinePattern

11.4.4.1 Class semantics

SY_LinePattern specializes SY_Presentation as an abstract line pattern. The line pattern has a coordinate reference system defined at every point along its path, which has an x-axis that is tangential to the path at that point. The line pattern also has a one dimensional coordinate reference system that is define along the path of the line pattern.

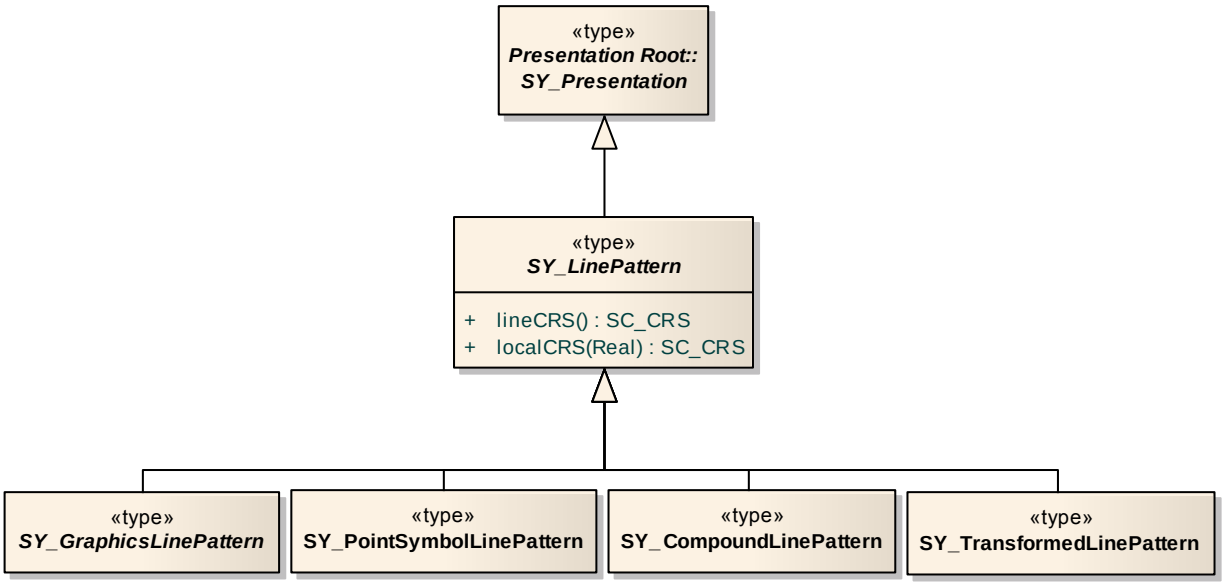

Figure 82 — SY_LinePattern context diagram

11.4.4.2 Generalization – SY_Presentation

SY_LinePattern specializes the abstract root type "SY_Presentation" as a an abstract line pattern. As such it shall implement all inherited attributes, operations and associations.

11.4.4.3 Operation – lineCRS

The operation "lineCRS" returns a one dimensional coordinate reference system that is define along the path of the line pattern.

```
SY_LinePattern::lineCRS(
  ) : SC_CRS
```

11.4.4.4 Operation – localCRS

The operation "localCRS" shall accept a real value as input and return a coordinate reference system. The parameter "measure" specifies the measure along the path of the line pattern specifying the origin of the coordinate reference system. The x-axis is tangential to the path at that point.

```
SY_LinePattern::localCRS(
     measure : Real
  ) : SC_CRS
```

11.4.5 Type – SY_CompoundLinePattern

11.4.5.1 Class semantics

SY_CompoundLinePattern specializes SY_LinePattern as a composition of subpatterns. The subpatterns may either be inline or referenced as portrayal features. The elements of the composition are ordered for display.

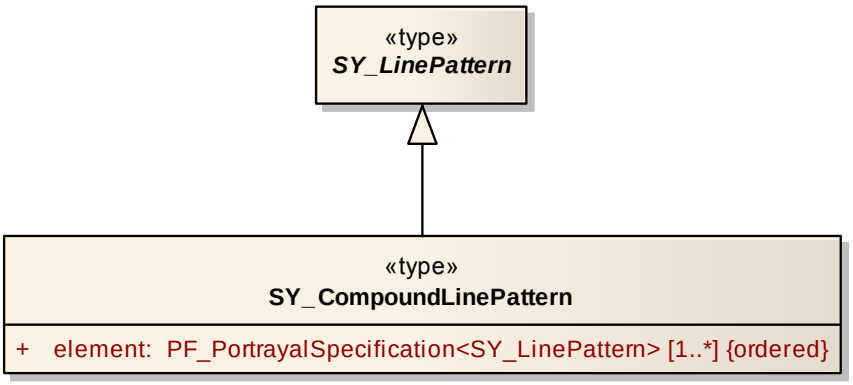

Figure 83 — SY_CompoundLinePattern context diagram

11.4.5.2 Generalization – SY_LinePattern

SY_CompoundLinePattern specializes the type "SY_LinePattern" as a compound of multiple subpatterns. As such it shall implement all inherited attributes, operations and associations.

11.4.5.3 Attribute – element

The attribute "element" specifies the ordered list of subpatterns of a compound line pattern.

```
SY_CompoundLinePattern::element :
     PF_PortrayalSpecification<SY_LinePattern>  [1..*]
```

11.4.6 Type – SY_TransformedLinePattern

11.4.6.1 Class semantics

SY_TransformedLinePattern specializes SY_LinePattern as a geometric transformation of the original pattern. The transformed line pattern may either be inline or referenced as a portrayal feature.

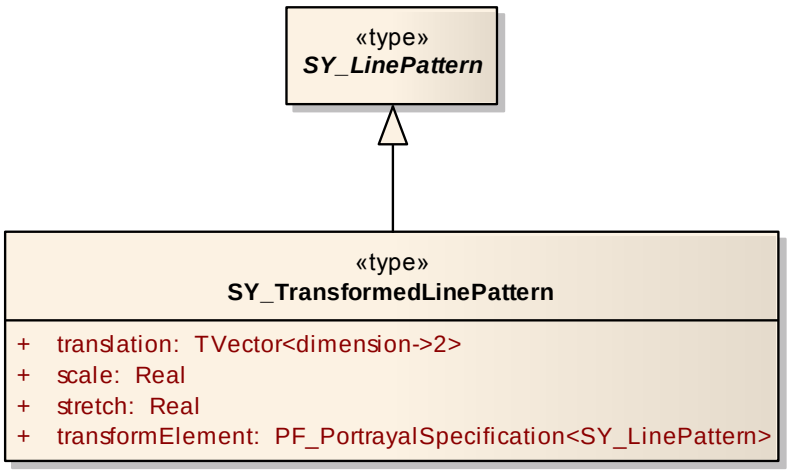

Figure 84 — SY_TransformedLinePattern context diagram

11.4.6.2 Generalization – SY_LinePattern

SY_TransformedLinePattern specializes the type "SY_LinePattern" as a geometrically transformed line pattern. As such it shall implement all inherited attributes, operations and associations.

11.4.6.3 Attribute – translation

The attribute "translation" specifies the translation of the pattern relative to the superpattern in two dimensions: along the path of the line pattern, and perpendicular to the path of the line pattern.

```
SY_TransformedLinePattern::translation : TVector<dimension->2>
```

11.4.6.4 Attribute – scale

The attribute "scale" specifies the scaling of the pattern perpendicular to the path of the line pattern.

```
SY_TransformedLinePattern::scale : Real
```

11.4.6.5 Attribute – stretch

The attribute "stretch" specifies the stretch of the pattern along the path of the line pattern.

```
SY_TransformedLinePattern::stretch : Real
```

11.4.6.6 Attribute – transformElement

The attribute "transformedElement" specifies the pattern to be transformed. The transformed pattern may either be inline or referenced as a portrayal feature.

```
SY_TransformedLinePattern::transformElement :
     PF_PortrayalSpecification<SY_LinePattern>
```

11.4.7 Type – SY_PointSymbolLinePattern

11.4.7.1 Class semantics

SY_PointSymbolLinePattern is a pattern of repeated point symbols along the path of the line pattern. The symbol is repeated at regular intervals and may have a masking, which masks out the underlying superpattern around the point symbol. For the point symbol not to be rendered at the unit length intervals but offset a line pattern transformation may be used.

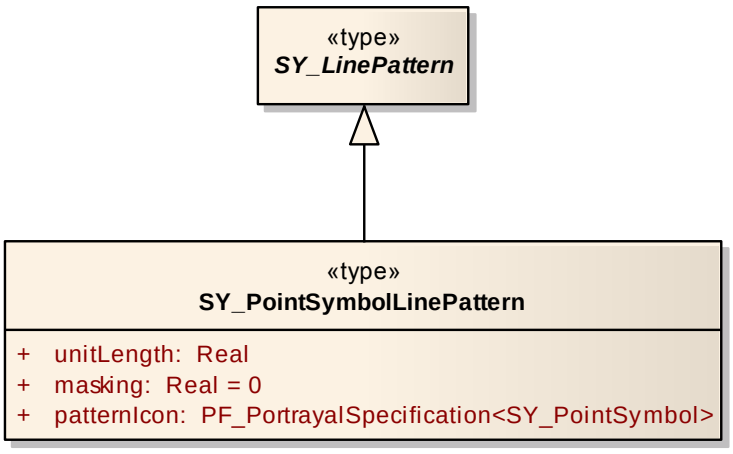

Figure 85 — SY_PointSymbolLinePattern context diagram

11.4.7.2 Generalization – SY_LinePattern

SY_PointSymbolLinePattern specializes the type "SY_LinePattern" as a pattern of repeated point symbols along the path of the line pattern. As such it shall implement all inherited attributes, operations and associations.

11.4.7.3 Attribute – unitLength

The attribute "unitLength" specifies the repetition interval length for the point symbol along the path of the line pattern. The interval is given in the coordinate reference system of the line pattern (operation "lineCRS").

```
SY_PointSymbolLinePattern::unitLength : Real
```

11.4.7.4 Attribute – masking

The attribute "masking" specifies the extent of the masking border masking out the underlying superpattern around the point symbol.

```
SY_PointSymbolLinePattern::masking : Real = 0
```

11.4.7.5 Attribute – patternIcon

The attribute "patternIcon" specifies the point symbol repeated along the path of the line pattern. The point symbol may either be inline or referenced as a portrayal feature.

```
SY_PointSymbolLinePattern::patternIcon :
     PF_PortrayalSpecification<SY_PointSymbol>
```

11.4.8 Type – SY_GraphicsLinePattern

11.4.8.1 Class semantics

SY_GraphicsLinePattern specializes SY_LinePattern as an abstract graphic element of a line pattern. This type is realized by a pen style in a defined graphics language.

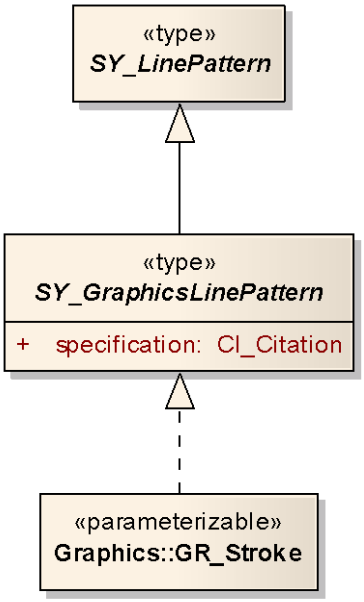

Figure 86 — SY_GraphicsLinePattern context diagram

11.4.8.2 Generalization – SY_LinePattern

SY_GraphicsLinePattern specializes the type "SY_LinePattern" as a graphic representation of the line pattern. As such it shall implement all inherited attributes, operations and associations.

11.4.8.3 Attribute – specification

The attribute "specification" cites the specification standard for the graphics definition language used to define the line pattern graphic.

```
SY_GraphicsLinePattern::specification : CI_Citation
```

11.5 Package – Area

11.5.1 Package semantics

The Area package specifies area symbols and area fills. An area symbol is composed of an area fill, a boundary line pattern, point symbols at specific locations within the area. An area fill is a fill pattern to render within an area. An area fill is composed of fill graphics and other area fills placed with a transformation. The graphics of the area fill use existing graphics standards.

The package contains 7 types. The type SY_AreaSymbol specifies an area symbol composed of an SY_AreaFill, a boundary pattern composed of an SY_LinePattern, and point symbols with in the area that are SY_PointSymbols. The type SY_AreaFill is the abstract root specialized by SY_GraphicsFill, SY_HatchFill, SY_PatternFill, SY_TransformedAreaFill, and SY_CompoundAreaFill. A SY_GraphicsFill is the graphic basis of an area fill. SY_HatchFill is a hatch fill using SY_LinePatterns for the hatching. SY_PatternFill fills an area with a pattern of SY_PointSymbols. SY_CompoundAreaFill composes multiple area fills into one, and SY_TransformedAreaFill transforms an area fill.

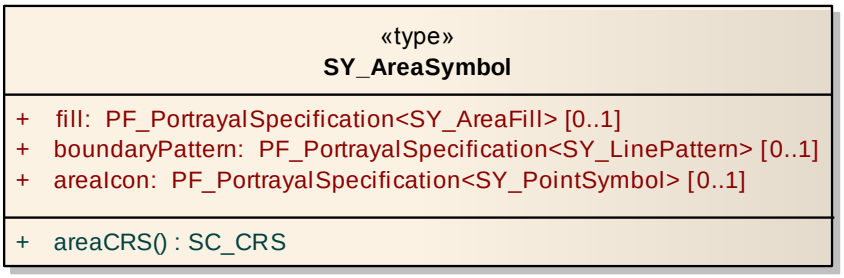

Figure 87 — Area Symbol

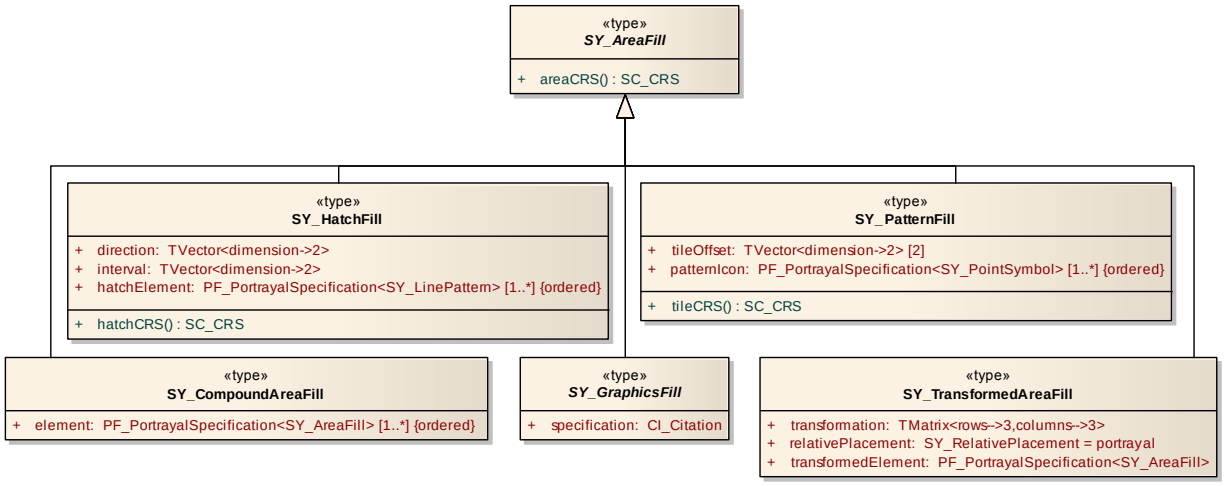

Figure 88 — Area Fill

11.5.2 Type – SY_AreaSymbol

11.5.2.1 Class semantics

SY_AreaSymbol specializes SY_Presentation as an area symbol that is composed of an area fill, a boundary line pattern, and point symbols at specific locations within the area. The area symbol has a coordinate reference system defined on the interior of the area.

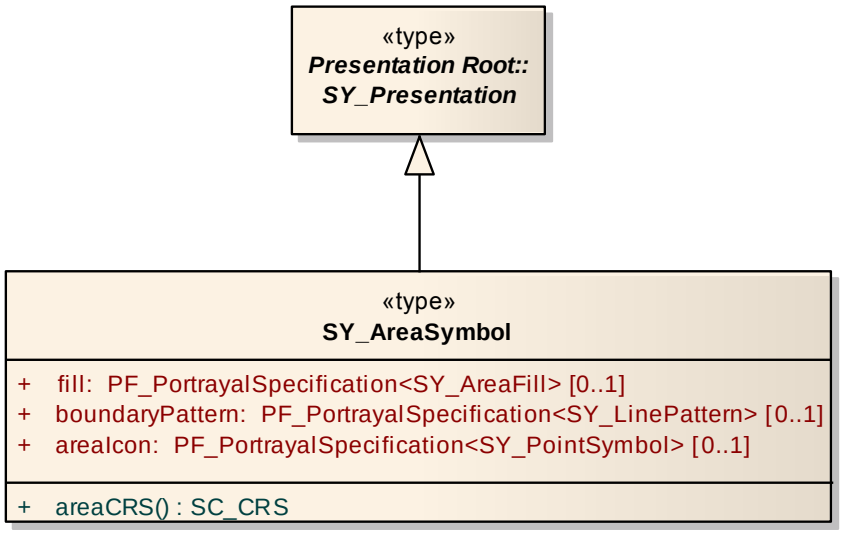

Figure 89 — SY_AreaSymbol context diagram

11.5.2.2 Generalization – SY_Presentation

SY_AreaSymbol specializes the abstract root type "SY_Presentation" as an area symbol. As such it shall implement all inherited attributes, operations and associations.

11.5.2.3 Attribute – fill

The optional attribute "fill" specifies the area fill of the area symbol.

```
SY_AreaSymbol::fill : PF_PortrayalSpecification<SY_AreaFill>  [0..1]
```

11.5.2.4 Attribute – boundaryPattern

The optional attribute "boundaryPattern" specifies the line pattern of the area symbol use to symbolize the area boundary.

```
SY_AreaSymbol::boundaryPattern :
     PF_PortrayalSpecification<SY_LinePattern>  [0..1]
```

11.5.2.5 Attribute – areaIcon

The optional attribute "areaIcon" specifies the point symbol used at specific locations as part of the area symbol use at specific locations within the area.

```
SY_AreaSymbol::areaIcon : PF_PortrayalSpecification<SY_PointSymbol>
     [0..1]
```

11.5.2.6 Operation – areaCRS

The operation "areaCRS" returns the coordinate reference system of the area symbol. The coordinate reference system is defined for the interior of the area.

```
SY_AreaSymbol::areaCRS(
   ) : SC_CRS
```

11.5.3 Type – SY_AreaFill

11.5.3.1 Class semantics

SY_AreaFill specializes SY_Presentation as an abstract area fill. The area fill has a coordinate reference system defined for the fill area.

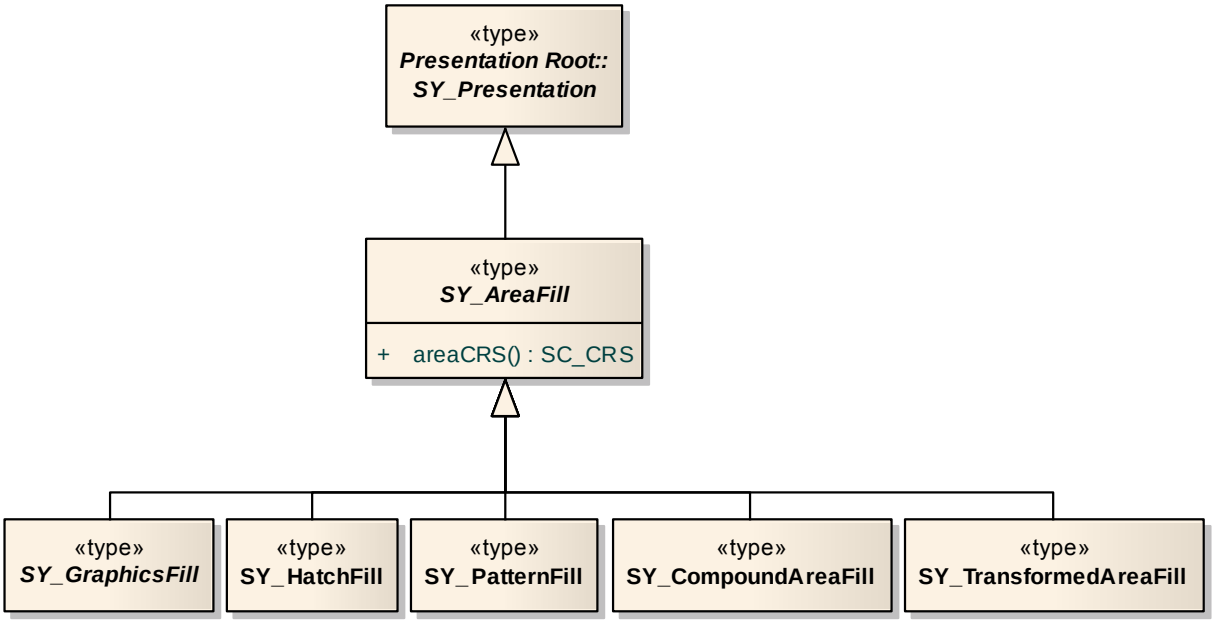

Figure 90 — SY_AreaFill context diagram

11.5.3.2 Generalization – SY_Presentation

SY_AreaFill specializes the abstract root type "SY_Presentation" as an area fill. As such it shall implement all inherited attributes, operations and associations.

11.5.3.3 Operation – areaCRS

The operation "areaCRS" returns a two dimensional coordinate reference system that is defined for the fill area.

```
SY_AreaFill::areaCRS(
   ) : SC_CRS
```

11.5.4 Type – SY_CompoundAreaFill

11.5.4.1 Class semantics

SY_CompoundAreaFill specializes SY_AreaFill as a composition of subfills. The subfills may either be inline or referenced as portrayal features. The elements of the composition are ordered for display.

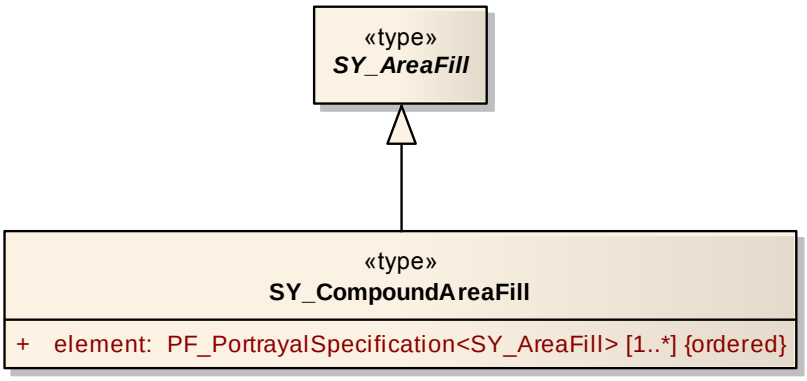

Figure 91 — SY_CompoundAreaFill context diagram

11.5.4.2 Generalization – SY_AreaFill

SY_CompoundAreaFill specializes the type "SY_AreaFill" as a compound of multiple subfills. As such it shall implement all inherited attributes, operations and associations.

11.5.4.3 Attribute – element

The attribute "element" specifies the ordered list of subfills of a compound area fill.

```
SY_CompoundAreaFill::element : PF_PortrayalSpecification<SY_AreaFill>
     [1..*]
```

11.5.5 Type – SY_TransformedAreaFill

11.5.5.1 Class semantics

SY_TransformedAreaFill specializes SY_AreaFill as a geometric transformation of the original fill. The transformed area fill may either be inline or referenced as a portrayal feature.

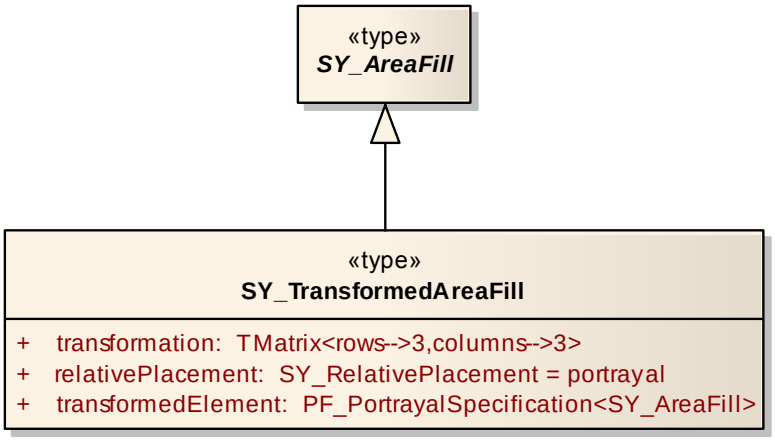

Figure 92 — SY_TransformedAreaFill context diagram

11.5.5.2 Generalization – SY_AreaFill

SY_TransformedAreaFill specializes the type "SY_AreaFill" as a geometrically transformed fill. As such it shall implement all inherited attributes, operations and associations.

11.5.5.3 Attribute – transformation

The attribute "transformation" uses homogeneous coordinates to define a geometric transformation of an area fill.

```
SY_TransformedAreaFill::transformation : TMatrix<rows→3,columns→3>
```

11.5.5.4 Attribute – relativePlacement

The attribute "relativePlacement" specifies if the transformation is relative to the superordinate presentation, relative to the portrayal coordinate reference system, or not applicable. The default value of this attribute is "portrayal".

```
SY_TransformedAreaFill::relativePlacement : SY_RelativePlacement =
     portrayal
```

11.5.5.5 Attribute – transformedElement

The attribute "transformedElement" specifies the fill to be transformed. The transformed fill may either be inline or referenced as a portrayal feature.

```
SY_TransformedAreaFill::transformedElement :
     PF_PortrayalSpecification<SY_AreaFill>
```

11.5.6 Type – SY_HatchFill

11.5.6.1 Class semantics

SY_HatchFill is a hatch fill created by filling an area with repeated application of a line pattern, applied in a specified direction, spaced evenly over a fill area, space with a specified interval.

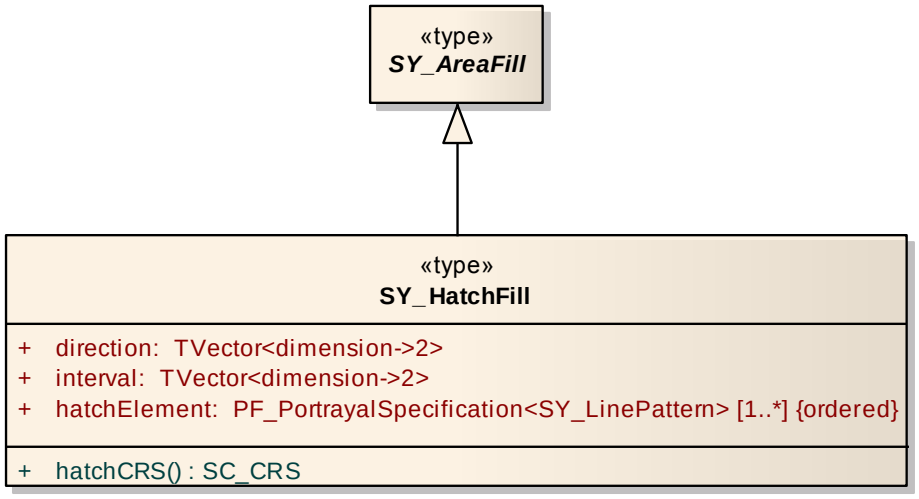

Figure 93 — SY_HatchFill context diagram

11.5.6.2 Generalization – SY_AreaFill

SY_HatchFill specializes the type "SY_AreaFill" as a hatch fill. As such it shall implement all inherited attributes, operations and associations.

11.5.6.3 Attribute – direction

The attribute "direction" specifies the direction of the hatch lines.

```
SY_HatchFill::direction : TVector<dimension->2>
```

11.5.6.4 Attribute – interval

The attribute "interval" specifies the interval between the hatch lines.

```
SY_HatchFill::interval : TVector<dimension->2>
```

11.5.6.5 Attribute – hatchElement

The attribute "hatchElement" specifies the line pattern applied to the hatch lines. The line pattern may either be inline or referenced as a portrayal feature.

```
SY_HatchFill::hatchElement : PF_PortrayalSpecification<SY_LinePattern>
     [1..*]
```

11.5.6.6 Operation – hatchCRS

The operation "hatchCRS" returns a one dimensional coordinate reference system that is define along the path of a hatch line.

```
SY_HatchFill::hatchCRS(
) : SC_CRS
```

11.5.7 Type – SY_PatternFill

11.5.7.1 Class semantics

SY_PatternFill is a fill created by filling an area with repeated application of point symbol tiles. The tiles are applied regularly at intervals defined by two vectors.

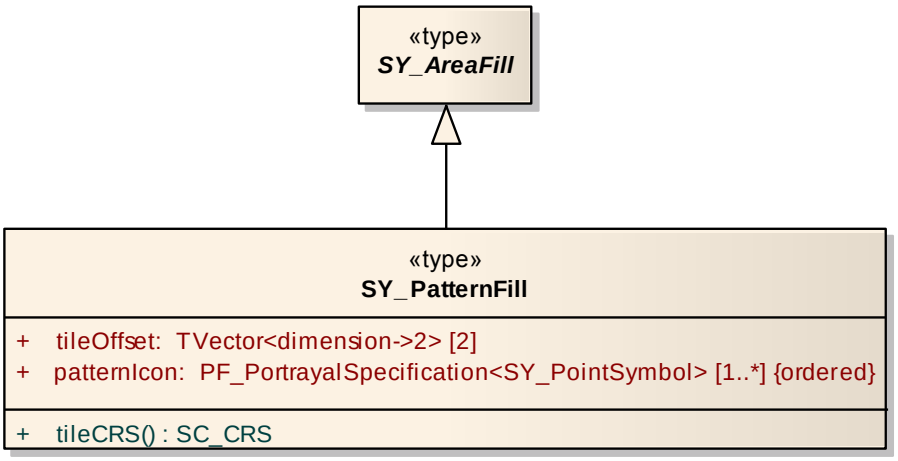

Figure 94 — SY_PatternFill context diagram

11.5.7.2 Generalization – SY_AreaFill

SY_PatternFill specializes the type "SY_AreaFill" as a pattern fill with a tile pattern of point symbols. As such it shall implement all inherited attributes, operations and associations.

11.5.7.3 Attribute – tileOffset

The attribute "tileOffset" specifies two vectors that describe the offsets of the tiles in two directions.

```
SY_PatternFill::tileOffset : TVector<dimension->2> [2]
```

11.5.7.4 Attribute – patternIcon

The attribute "patternIcon" specifies the ordered point symbols in a tile of a pattern fill. The symbols are placed within the tile coordinate reference system. The point symbol may either be inline or referenced as a portrayal feature.

```
SY_PatternFill::patternIcon : PF_PortrayalSpecification<SY_PointSymbol>
     [1..*]
```

11.5.7.5 Operation – tileCRS

The operation "tileCRS" returns the coordinate reference system for a tile of the area fill.

```
SY_PatternFill::tileCRS(
   ) : SC_CRS
```

11.5.8 Type – SY_GraphicsFill

11.5.8.1 Class semantics

SY_GraphicsFill specializes SY_AreaFill as an abstract graphic element of an area fill. This type is realized by a fill in a defined graphics language.

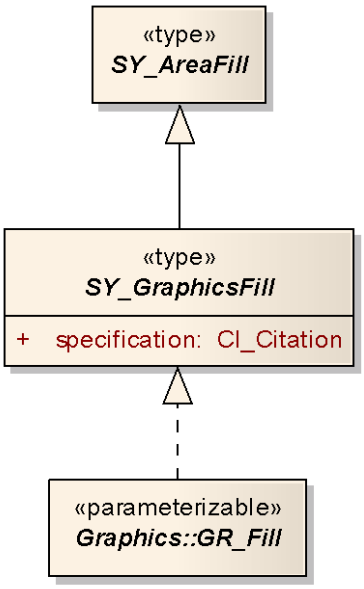

Figure 95 — SY_GraphicsFill context diagram

11.5.8.2 Generalization – SY_AreaFill

SY_GraphicsFill specializes the type "SY_AreaFill" as a graphic representation of an area fill. As such it shall implement all inherited attributes, operations and associations.

11.5.8.3 Attribute – specification

The attribute "specification" cites the specification standard for the graphics definition language used to define the area fill graphic.

```
SY_GraphicsFill::specification : CI_Citation
```

11.6 Package – Text

11.6.1 Package semantics

The Text package specifies text symbols for locations and along paths with 3 types: SY_TextSymbol, SY_GraphicsTextPoint, and SY_GraphicsTextCurve. SY_TextSymbol is the abstract root for SY_GraphicsTextPoint and SY_GraphicsTestCurve. SY_GraphicsTextPoint places a text at a location. SY_GraphicsTextCurve renders a text along a path.

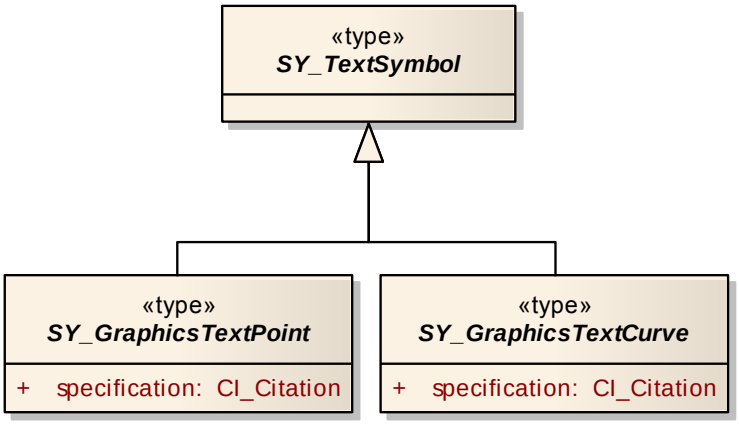

Figure 96 — Text Symbol

11.6.2 Type – SY_TextSymbol

11.6.2.1 Class semantics

SY_TextSymbol specializes SY_Presentation as an abstract text symbol.

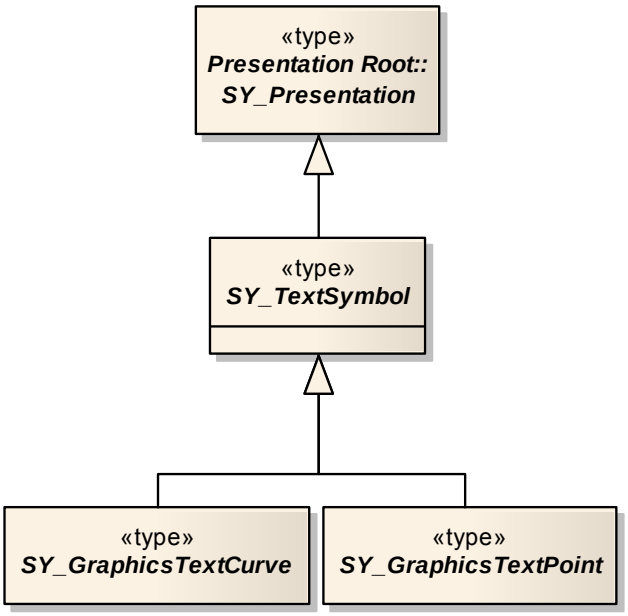

Figure 97 — SY_TextSymbol context diagram

11.6.2.2 Generalization – SY_Presentation

SY_TextSymbol specializes the abstract root type "SY_Presentation" as a text symbol. As such it shall implement all inherited attributes, operations and associations.

11.6.3 Type – SY_GraphicsTextPoint

11.6.3.1 Class semantics

SY_GraphicsTextPoint specializes SY_TextSymbol as an abstract graphic point text. This type is realized by a text graphic in a defined graphics language.

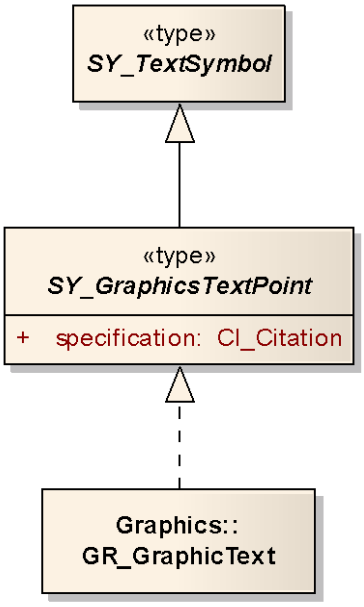

Figure 98 — SY_GraphicsTextPoint context diagram

11.6.3.2 Generalization – SY_TextSymbol

SY_GraphicsTextPoint specializes the type "SY_TextSymbol" as a graphic representation of a text graphic. As such it shall implement all inherited attributes, operations and associations.

11.6.3.3 Attribute – specification

The attribute "specification" cites the specification standard for the graphics definition language used to define the point text graphic.

```
SY_GraphicsTextPoint::specification : CI_Citation
```

11.6.4 Type – SY_GraphicsTextCurve

11.6.4.1 Class semantics

SY_GraphicsTextCurve specializes SY_TextSymbol as an abstract graphic path text (i.e. text that follows a path). This type is realized by a path text in a defined graphics language.

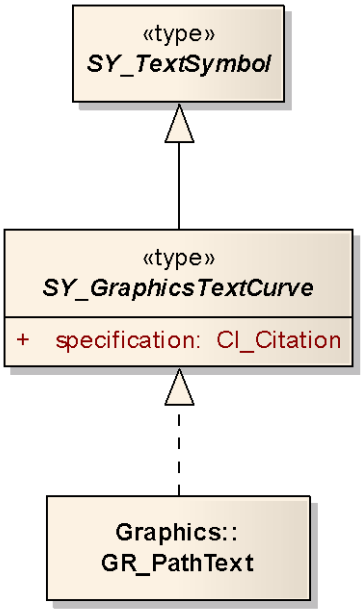

Figure 99 — SY_GraphicsTextCurve context diagram

11.6.4.2 Generalization – SY_TextSymbol

SY_GraphicsTextCurve specializes the type "SY_TextSymbol" as a graphic representation of a path text. As such it shall implement all inherited attributes, operations and associations.

11.6.4.3 Attribute – specification

The attribute "specification" cites the specification standard for the graphics definition language used to define the curve text graphic.

```
SY_GraphicsTextCurve::specification : CI_Citation
```

11.7 Other

11.7.1 Body/Solid

TBD

11.7.2 Diagram

Pass through from portrayal feature

11.7.3 Animation

Could be point symbol video clip, but also changing area symbol (e.g. Roman Empire expand and contract), or also line symbol (e.g. bus running up and down road, or animated progress of road being built)

11.7.4 Tactile

11.7.5 Aural

Annex A– Abstract test suite (normative)

TBD

Annex B – Functions – Profile of FE (normative)

TBD

Annex C – Expressions – Profile of FE (normative)

TBD

Annex D– Graphics [GR_] (normative)

D.1 UML

D.1.1 Package semantics

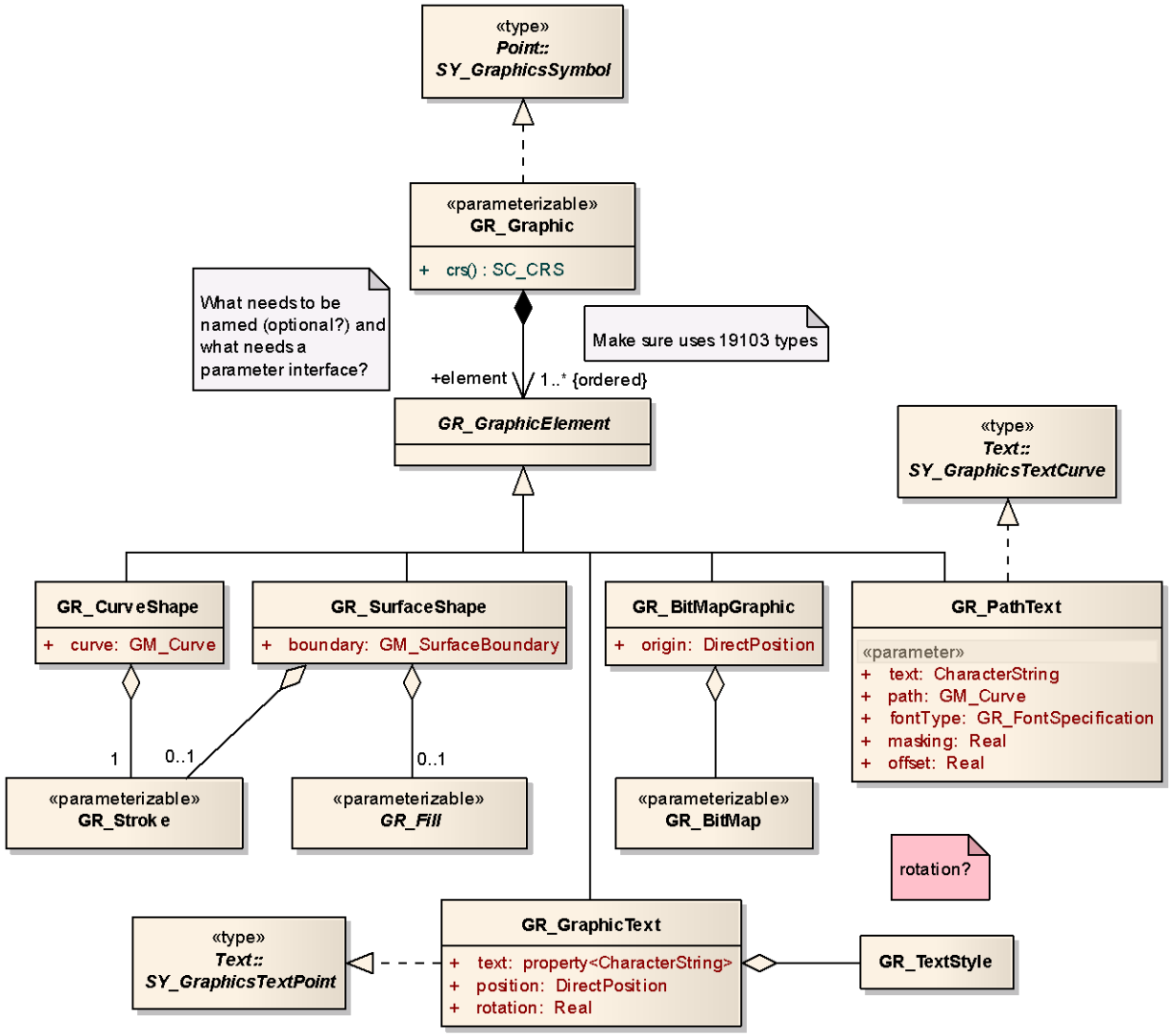

Figure 100 — Graphic

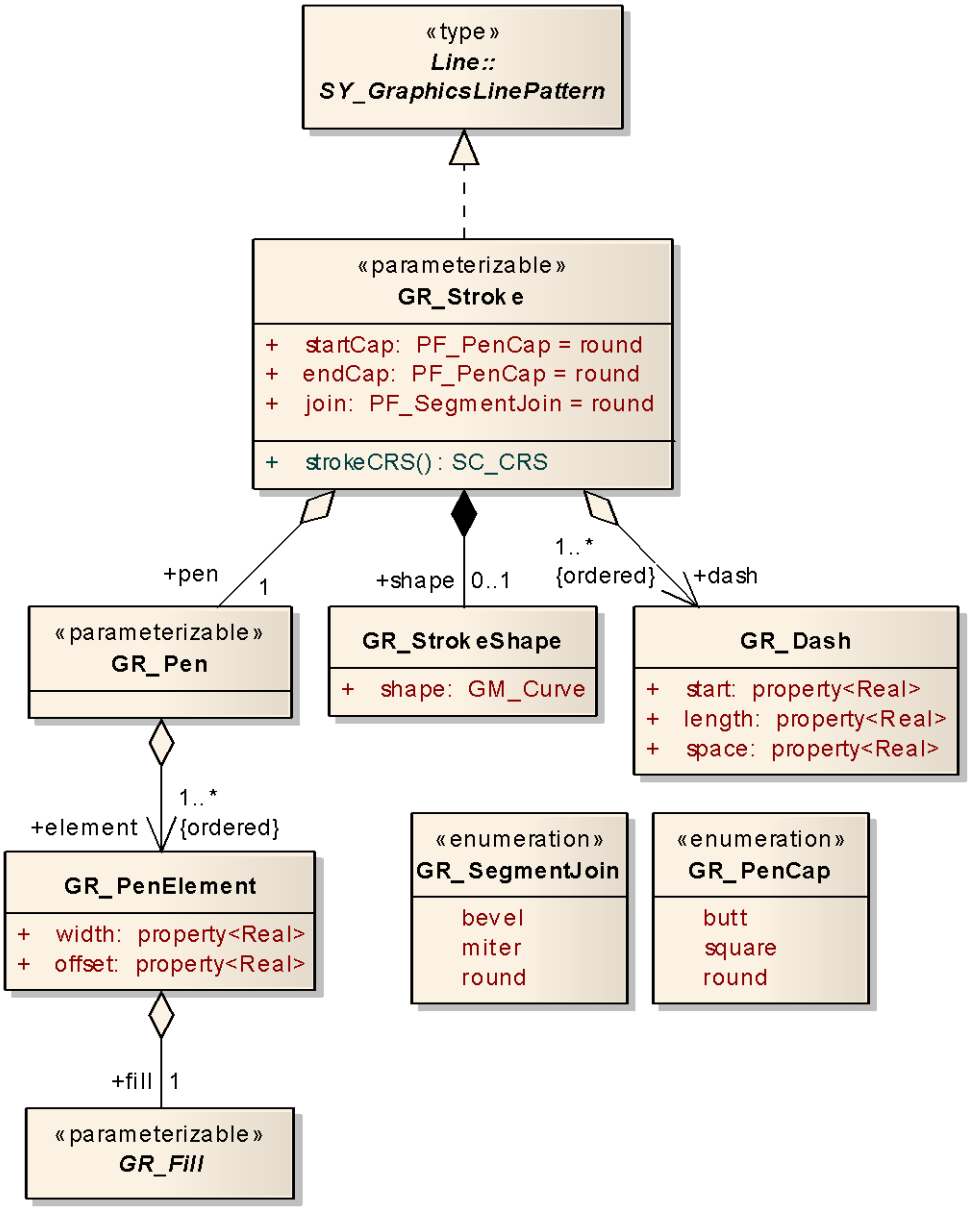

Figure 101 — Stroke

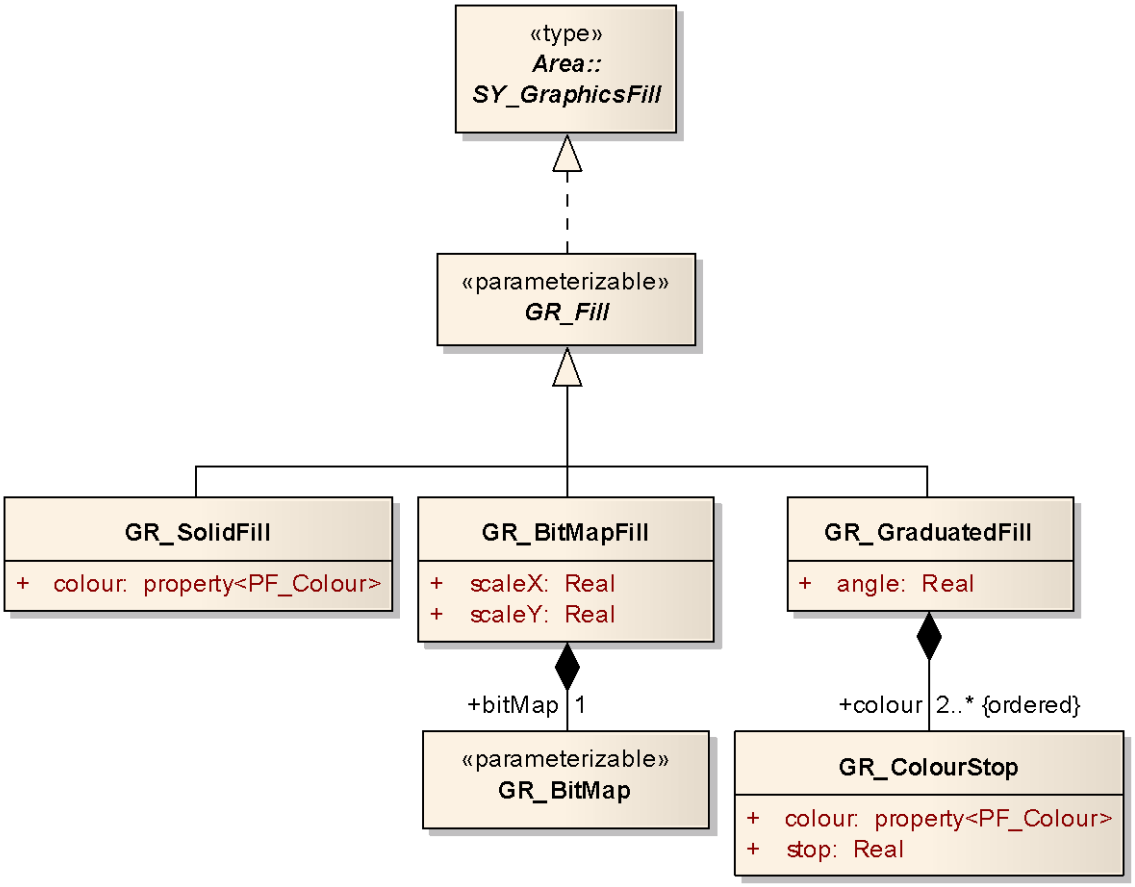

Figure 102 — Fill

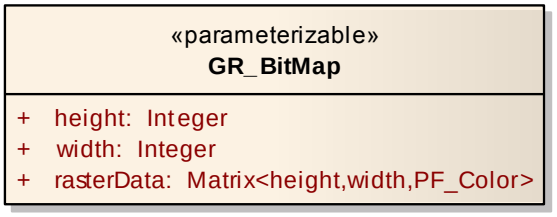

Figure 103 — BitMap

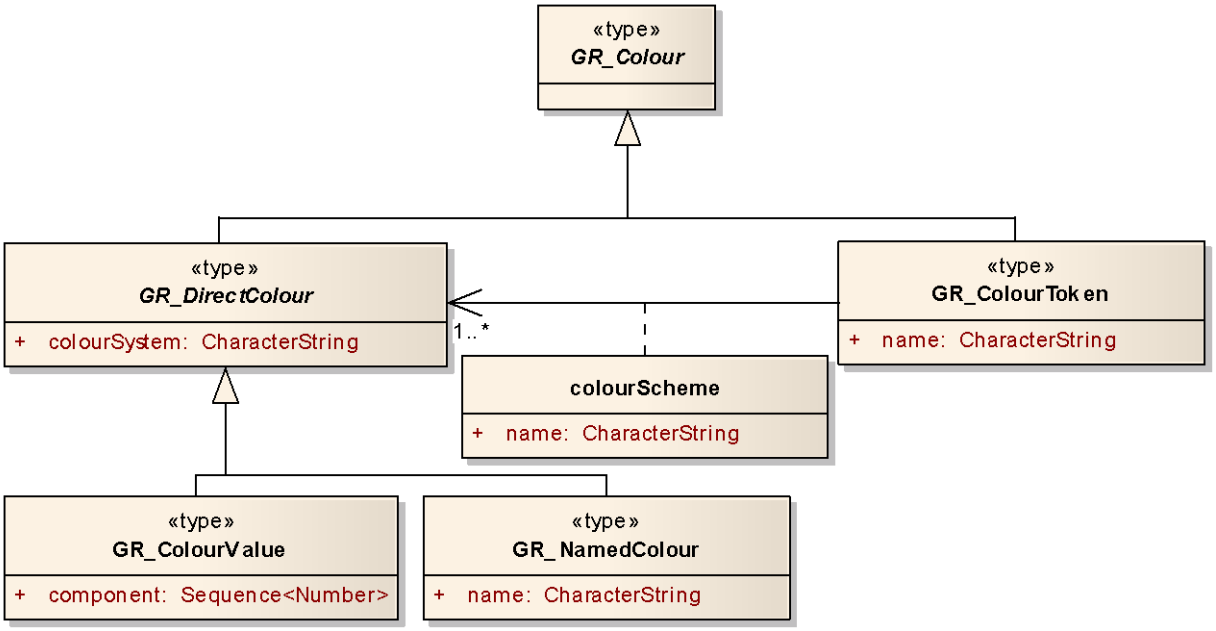

Figure 104 — Colour

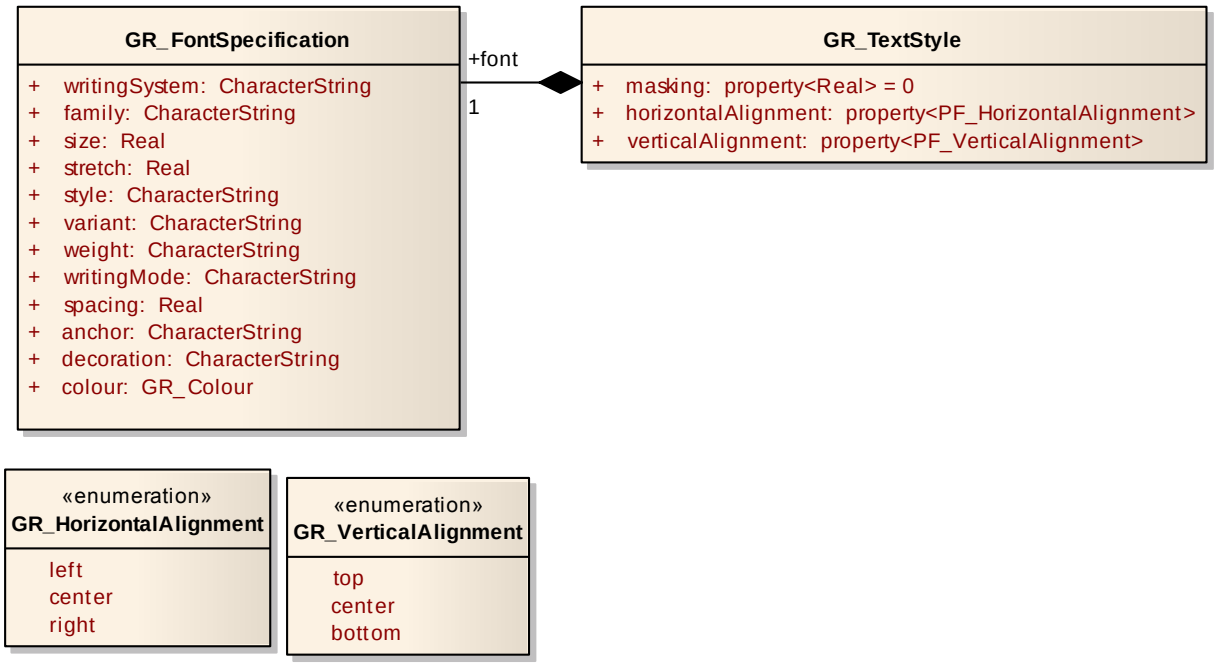

Figure 105 — Text Style

D.1.2 Package – Parameterization

D.1.2.1 Package semantics

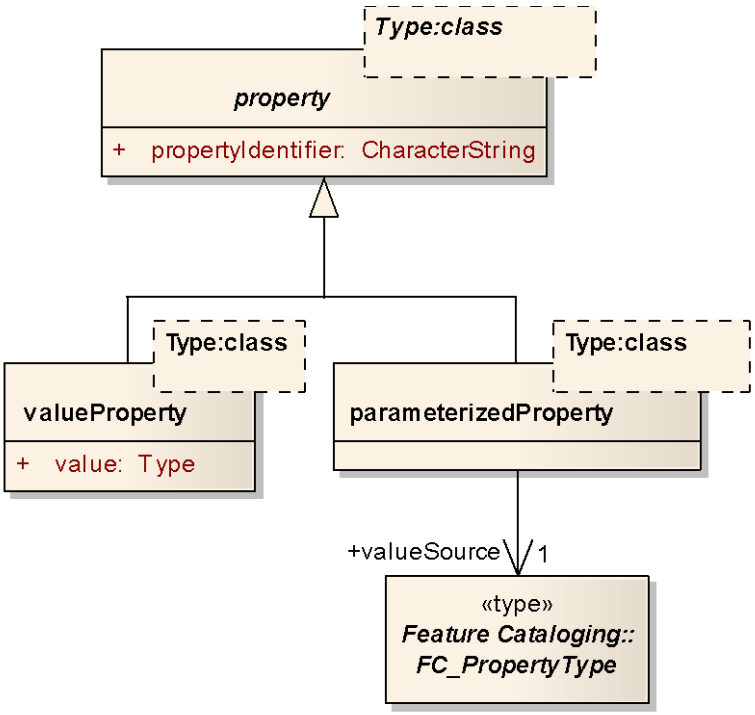

Figure 106 — Parameterization

D.1.2.2 – property

D.1.2.2.1 Class semantics

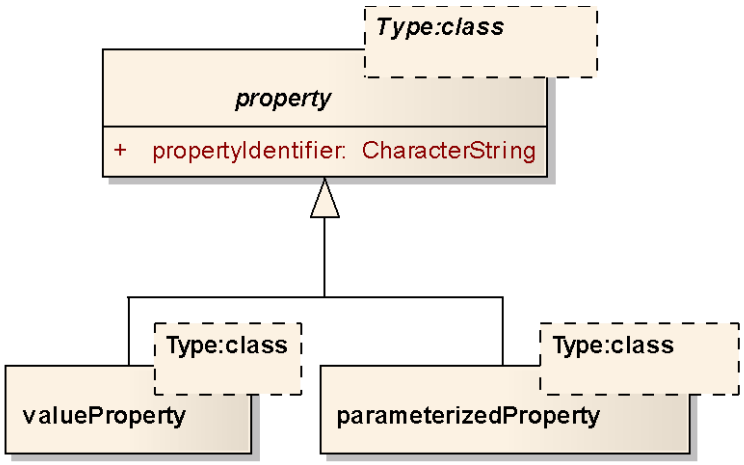

Figure 107 — property incomplete context diagram

D.1.2.3 – valueProperty

D.1.2.3.1 Class semantics

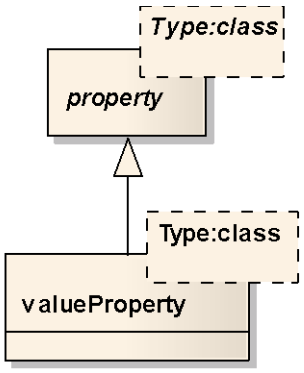

Figure 108 — valueProperty incomplete context diagram

D.1.2.4 – parameterizedProperty

D.1.2.4.1 Class semantics

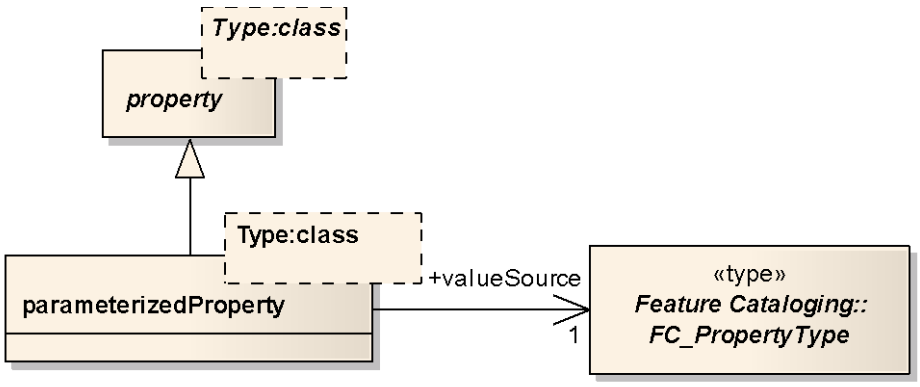

Figure 109 — parameterizedProperty incomplete context diagram

D.2 XML

TBD

Annex E – Mapping of ISO 19117 to OGC Symbology Encoding

TBD

Annex F – XML schema

TBD

Bibliography

[1] Gill, A.: *Applied Algebra for the Computer Sciences*. Series in Automatic Computation, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1976