Open Geospatial Consortium

Date: 2011-11-17

Reference number of this document: OGC 11-062r2

Category: Public Engineering Report

Editors: David Burggraf, Ron Lake

OWS-8 CCI Portrayal Registries Engineering Report

Warning

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. This document is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements.

Document type: OpenGIS® Engineering Report
Document subtype: NA
Document stage: Approved for public release
Document language: English

Preface

This document is a deliverable for the OGC Web Services 8 (OWS-8) testbed activity. OWS testbeds are part of OGC's Interoperability Program, a global, hands-on and collaborative prototyping program designed to rapidly develop, test and deliver proven candidate standards or revisions to existing standards into OGC's Standards Program, where they are formalized for public release. In OGC's Interoperability Initiatives, international teams of technology providers work together to solve specific geoprocessing interoperability problems posed by the Initiative's sponsoring organizations. OGC Interoperability Initiatives include test beds, pilot projects, interoperability experiments and interoperability support services - all designed to encourage rapid development, testing, validation and adoption of OGC standards.

The OWS-8 sponsors are organizations seeking open standards for their interoperability requirements. After analyzing their requirements, the OGC Interoperability Team recommend to the sponsors that the content of the OWS-8 initiative be organized around the following threads:

* Observation Fusion

* Geosynchronization (Gsync)

* Cross-Community Interoperability (CCI)

* Aviation

More information about the OWS-8 testbed can be found at:

http://www.opengeospatial.org/standards/requests/74

OGC Document [11-139] "OWS-8 Summary Report" provides a summary of the OWS-8 testbed and is available for download:

https://portal.opengeospatial.org/files/?artifact_id=46176

License Agreement

Contents

Page

OWS-8 CCI Portrayal Registries Engineering Report

1 Introduction

1.1 Scope

The OWS-8 Cross Community Interoperability (CCI) thread was built on progress made in the recent OWS-7 initiative to cover key technology areas that could not be addressed within the scope of that initiative. The OWS-8 CCI thread aimed to increase interoperability within communities sharing geospatial data, including advancing of interoperability among heterogeneous data models, advancing strategies to share styles to provide a more common and automated use of symbology, improvement of KML, and advancing schema automation allowing communities to better share their information artifacts. This OGC engineering report aims to present findings from the portrayal registries as part of the CCI subthread

These guidelines are drawn from lessons learned in the framework of the OWS-8 initiative environment, which includes multiple OGC service implementations including WMS/FPS (Carmenta), WFS (CubeWerx, Luciad, interactiveInstruments), CSW-ebRIM for Services (Compusult), CSW ebRIM Portrayal Registry (Carmenta), web portal client applications (ESRI and Compusult). The following was the scope of the work, related to Portrayal performed at the CIC thread:

- Develop a CSW ebRIM Profile for Portrayal Registry.
- Develop a CSW ebRIM interface to the DGIWG Portrayal Registry (to evolve into portrayal extension package of CSW ebRIM).
- Advance the use of SLD documents by creating a link between a SLD document in a Get Map Request and SE from the portrayal registry CSW ebRIM interface (Reference OGC SLD Profile of WMS Change Proposal 10-181). Intent for this extension point is to allow a reference to a rule set ID in a portrayal registry to be included in a SLD document.
- Evaluate and demonstrate if the same style resources could be used for OGC Portrayal Services and KML data, i.e. KML and SLD/SE would simply be different representations of the same resource.
- Evaluate the use of portrayal registries within KML. KML features reference styles, while OGC Portrayal Service requests reference SLD (containing a set of layers with one or more styles). These resources would need to be managed consistently in the registry.
- Improve the portrayal registry service by allowing styles to be accessed from the registry using a URL with a fragment identifier to reference the particular Style/StyleMap element in the KML document.
- Improve the portrayal registry service by making the URL above mentioned to be persistent - or available at least as long as anyone might reference the particular style(s).

- Identify need for extension points to support military symbology, including MIL-DTL-89045A, GeoSym and MIL-STD-2525C, Common Military Symbology.

- Develop a small sample set of symbols (SLD/SE and KML) for demonstration purposes.

1.2 Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Name	Organization
David Burggraf	Galdos Systems
Ron Lake	Galdos Systems
Tobias Moberg	Carmenta
Mark Simms	Compusult
Peter Vretanos	Cube Werx
Luis Bermudez	Open Geospatial Consortium
Clemens Portele	Interactive Instruments
Daniel Balog	Luciad

1.3 Revision history

Date	Release	Editor	Primary clauses modified	Description
21-06-2011	0.1	David Burggraf	All	First draft
14-08-2011	0.2	Ron Lake	All	Second draft
18-08-2011	0.3	Ron Lake	All	Third draft – Portrayal Registry Model Comparison
28-08-2011	0.4	Ron Lake	All	Fourth draft
03-09-2011	0.5	Ron Lake	All – added material on KML styling.	Fifth draft
14-09-2011	0.6	Ron Lake	Expanded section on KML Styling, Revised recommended extension pkg model. Added Bibliography.	
22-09-2011	0.7	Luis Bermudez	All, particular 1.1 and 4.1	General edits, edited scope and introduction for OWS-8 and CCI
29-09-2011	0.8	David Burggraf	All	Overall Review

1.4 Future work

It is suggested that this report form the basis for a Working Group to develop a CSW-ebRIM extension package for a Portrayal Registry.

1.5 Forward

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

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

2 References

The following documents are referenced in this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For un-dated references, the latest edition of the normative document referred to applies.

LTDS_Symbology_V.0.2_draft_30_Sept_2010.pdf
Draft NSG, Portrayal Standard for LTDS Data

3 Conventions

3.1 Abbreviated terms

AIXM	Aeronautical Information (Ex)change Model
CSW-ebRIM	Catalogue Service for the Web, ebRIM profile.
DGIWG	Defense Geospatial Information Working Group
ebRIM	e-Business Registry Information Model
FPS	Feature Portrayal Service
GML	Geography Markup Language
ISO	International Organization for Standardization
KML	Earth browser language, formerly Keyhole Markup Language
LTDS	Local Topographic Data Store
OASIS	Organization for the Advancement of Structured Information Standards
OGC	Open Geospatial Consortium
PR	Portrayal Registry
SE	Symbology Encoding
SLD	Styled Layer Descriptor
SVG	Scalable Vector Graphics
WFS	Web Feature Service
WMS	Web Map Service
WPS	Web Processing Service
XML	Extensible Markup Language

4 Portrayal Registry Models

4.1 Introduction

While the concept of a Portrayal Registry is quite long standing, the formalization of the concept into a robust standards specification has been elusive. The difficulties in creating a Portrayal Registry standard should not be underestimated, and include technical, conceptual and even "cultural" issues. To understand this we need only recall that while the first Feature Portrayal Service was demonstrated in OGC Web Mapping Test Bed 1 (WMT1), a formal "FPS" specification was not written until 2006. It took that long to understand the differences between FPS and conventional ("legacy") map servers, and to reach agreement on at least a basic model of the portrayal process. Note further that while ISO TC/211 19117, was one of the earliest specifications to be discussed jointly between the OGC and ISO TC/211, there is still, even today, no broad agreement on a suitable reference model for portrayal.

The Portrayal Registry component of OWS7 showed clearly that the OGC CSW-ebRIM specification was an appropriate foundation on which to build a Portrayal Registry, however, it did not result in a portrayal registry, i.e. ebRIM extension package ready for standardization. It is hoped that the work in this component, when combined with that of other threads making use of ebRIM registries will at least provide a concrete starting point.

One of the objectives of this report is to further illuminate the issues involved with the hope of laying the ground work for a formal Portrayal Registry specification that can become both an OGC and ISO TC/211 standard. In particular, the Portrayal Registry prototype should be built on top of the DGIWG portrayal database. This database does not use SE as its internal representation; it is based on the portrayal concepts in ISO 19117 and adds some registration aspects from ISO 19135.

One important difference between the DGIWG database and the SE structure is that it separates rules from the symbols. This separation is actually quite important for the DGIWG use case; it is envisioned that the nations in DGIWG will agree on a common set of symbols for (say) topographical maps, and that individual member nations will then create rule sets which map these symbols to the (potentially nation-specific) application schemas. (Note that symbols refer to here include icons, line styles, text styles and area fills.)

4.2 Overview of Portrayal Process

This section provides an informal discussion of the Portrayal Process. It is not specifically based on either OGC SLD/SE, nor on ISO 19117, but shares elements of both.

The intent of the Portrayal Process is to generate graphical renditions based on geographic or geographically related data. In most cases this means the generation of maps, however, other kinds of visual products may be produced. Whether the graphical rendition is expressed on paper or delivered only on electronic media will not concern us here, other than to note that the introduction of electronic media has greatly extended the possibilities of portrayal, including everything from dynamic highlighting and labeling (e.g. mouse overs), to the generation of photo-realistic 3D models of buildings and entire cities.

In portrayal we assume that we are provided with one or more sources of geographic or geographically related data, meaning some collection of encodings of models of real world objects, and it is the purpose of portrayal to use this information to create an appropriate graphical rendition. So our simplest model of portrayal is as shown in Figure 1.

Figure 1: Very Simple Model of Portrayal

The graphical world is distinguished from the model of the real world. We separate the presentation (portrayal) from the real world model (content). Typically we have different means for encoding models of real world objects, than for the description of the graphical objects. We also avoid the use of terminology like “conversion” since the graphical rendition is not a conversion of the geographic world, but rather an interpretation or styling. The process of interpreting the encodings of models of real world objects to create graphical objects is called styling, and the process of transforming the graphic object encodings into something we see on a screen or paper, is called rendering. Portrayal is then the composition of Styling and Rendering.

A canonical example of Figure 1 is the case of encoding models of real world objects in GML, styling them to SVG (scalable vector graphics) and then rendering the SVG on a monitor screen or piece of paper. The difficulty of the whole subject of standardizing portrayal is underscored by the fact that we might not have universal agreement on the meaning of Figure 1, or that it is the essence of portrayal. Nonetheless this report will proceed on that assumption.

To refine the notion of Portrayal further we need to introduce a way of thinking about “models of real world objects”, and for that we will rely on ISO TC/211, 19109, which introduces the notion of features or application objects. This is another slippery slope, and we will find very quickly that we will want to distinguish certain kinds of features, we call coverages, while other, discrete objects, we are content to call just, features. Since this may be confusing, the term “discrete feature” will be used to refer to discrete objects (concrete or abstract) such as roads, rivers, buildings, electoral districts or air spaces. The term coverage will be used to refer to features which describe the distribution of some quality or quantity over a geographic region and/or in time. Coverages include things like sea surface temperature, population distributions, bore hole rock extrusions, salinity profiles in the ocean, or temperature profiles in the atmosphere. We should not be misled by the fact that the geometry of discrete features are most often encoded in so called “vector” descriptions, while coverages are often described using raster or gridded data structures. The geometry of discrete features could be described also using raster or gridded structures (e.g. bit maps or characteristic functions), while many kinds of coverages naturally arise from vector structures (e.g. distribution of birth rate by country in state of Alabama).

In ISO 19109, features are typed objects (the type has a name) that are characterized by a list of properties, the late 20th century incarnation of structuralist philosophy from earlier in the century.

As features are our means of describing the real world, so too, some set of graphical objects are used to describe what it is that appears in our graphical rendition. Typically such graphical objects are geometric entities and text, together with visual properties such as colour, reflectance, etc. The process of styling can then be seen as one of selecting and specifying the graphical objects that will represent particular feature objects.

Styling is thus the application of rules or functions that map feature objects to graphic objects.

F: feature |------------> graphic object

The selection and specification of graphical objects (styling) is a made more complex by several factors, including:

1. The geometry of the graphical objects may be strictly related (e.g. scaled) or strictly unrelated (e.g. map of the London underground) to the geometry of the geographic entities or features.
2. The graphical objects resulting from styling may represent the geographic features symbolically, in which case the geometry of the graphical objects may be radically different than that of the corresponding geographic features (e.g. consider a symbol representing the location of a fire).
3. Properties of the graphical objects may very often be used to express the properties of the geographic features, even non-geometric ones. For example, an earthquake feature may properties location, magnitude and type (.e.g. shallow fault, subduction etc), and the corresponding graphical object may be a circle, with center at the "location" of the earthquake, a diameter determined by the magnitude, and a colour represents the earthquake's type.
4. Features may have very detailed and very accurate geometric descriptions in 2, 3 or more dimensions, which are identified by their appropriate geometric properties. We thus speak about the centerline of a road segment (a curve) or the extent of a road segment (a polygon). The graphical objects resulting from styling might in this case be only a curve (for the road centerline) with a line weight indicating the extent, or they might be polygons. Styling a given feature object, or objects could thus result in something which is a "realistic" visual rendition, one that is partly realistic, or one that is entirely symbolic.
5. The point of portrayal is to communicate information. This is why we often mix both symbolic and realistic renditions. In conventional cartography, the importance of communication is well understood, and the graphical objects appearing on a map are quite often modified to ensure readability. For example, contour lines may be distorted, or roads moved so as to minimize visual interface between the objects in the rendition. This is not geographically correct, but it is better understood by the human reader.
6. Most graphical renditions benefit from the use of text, either as components of geometric objects or just as labels placed on their own. The size, content, orientation, colour etc. of this text usually depends (or not!) on the properties of the related feature

objects. Where this text is placed can be quite problematic. Again, readability issues play a major role.

7. A graphical rendition usually includes a “map” (collection of graphical objects) together with various auxiliary information pertinent to the interpretation or understanding of the “map”. This can include information such as scale, legends, and shading algorithms, as well as more conventional metadata like the authoring organization, and the date of publication.
8. Since styling is a process of interpretation, the graphical objects (as already seen) may be quite different than the corresponding feature objects.

4.3 Portrayal Registry Issues

The Portrayal Registry will be based on the OGC CSW-ebRIM specification, meaning that it is defined by an ebRIM extension package for portrayal. Thus all discussion of registry objects means ebRIM registry objects.

While the idea of a Portrayal Registry is quite old, there are many basic issues which have not been resolved sufficiently to define a portrayal registry model for a standards specification. These include:

9. **Content**: Should the registry include only symbols or should it include symbols and styles (how should style be defined?), and even rendition layer definitions.
10. **Purpose**: Should the registry’s purpose only be to support the 1) sharing/discovery of symbol (and style? and rendition layer definitions), or 2) construction of styles from style components (such as symbols).
11. **Granularity**: How fine grained should the objects in the registry be?
12. **Degree of Abstraction**. Since we have encodings for symbols (e.g. OGC SE, SVG, KML etc), how much of the symbol encoding should be exposed through the registry objects, and their properties.
13. **Information Hiding:** Can we structure the registry objects so that they can hide a number of different encodings with a common set of abstractions? For example, we might have an ExtrinsicObject for a styling rule, with the associated repository item holding an encoding of the rule, the nature of the encoding being specified by a slot on the ExtrinsicObject.

4.4 Extension Package for Portrayal from OWS-7

In the aviation thread of OWS-7 an extension package was defined with the following Registry Objects:

Class	Description	ebRIM Registry Object	Repository Item
Dataset	Geospatial dataset	ExtrinsicObject	Not specified
Data Product Specification	Definition of a data product.	ExtrinsicObject	Not specified
Data Product Portrayal	Collection of styling rules, bound to set of feature types	ExtrinsicObject	SLD document

Symbol Library	A collection of point, line, area, text symbols.	ExtrinsicObject	Unclear – suggested could be a GML dictionary.
Symbol	Abstract Class for Symbol. Entry in a symbol library	Classification Node	Not applicable
Vector Symbol	Vector Symbol encoded in SVG	ExtrinsicObject	Symbol in SVG
Raster Symbol	Raster symbol encoded in a bit map.	ExtrinsicObject	Bit map
LineStyle	Abstract class for line style.	Classification Node	Not applicable
VectorLineStyle	Class representing a line style encoded in SVG	ExtrinsicObject	LineStyle encoded in SVG
RasterLineStyle	Class representing a line style encoded as a bit map.	ExtrinsicObject	LineStyle encoded in a bit map.
FillStyle	Abstract class for a fill style	Classification Node	Not applicable
VectorFillStyle	Class representing a vector fill style encoded in SVG	ExtrinsicObject	VectorFill encoded in SVG
RasterFillStyle	Class representing a raster fill style encoded in SVG	ExtriniscObject	Raster fill encoded in SVG

While it was not done in OWS-7, one could add a slot to the ExtrinsicObjects to indicate the encoding of the associated repository item. For example, in the OWS-7 Portrayal Registry, this slot would have the value “SVG” for vector symbols, and “Bit Map” (or a specific bit map encoding) for raster symbols.

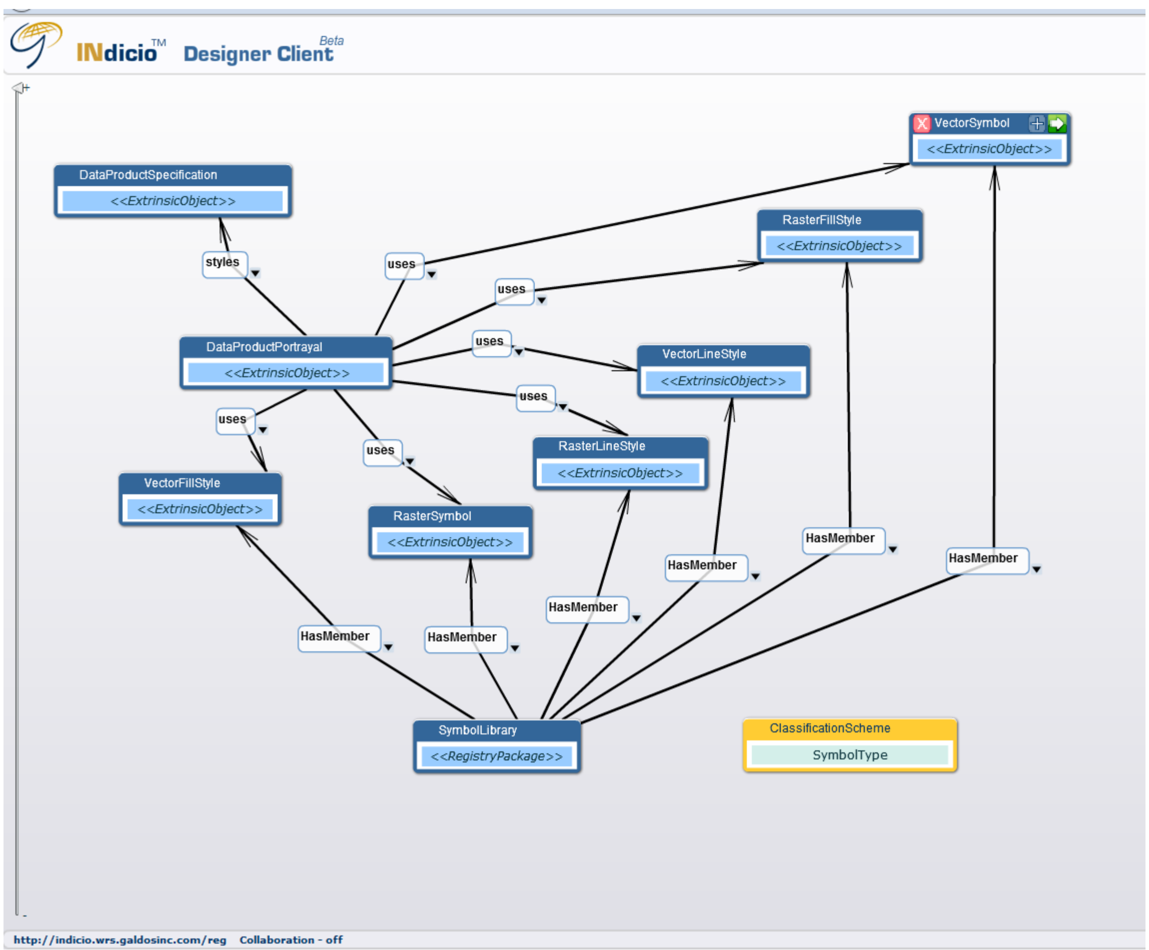


Figure 2. Portrayal Extension Package from OWS-7

4.5 Portrayal Registry Model based on OGC SLD/SE

The OGC SLD 1.1 and SE 1.1 specifications extended the portrayal model to more clearly support the concept of a Feature Portrayal Service or so called "Component" WMS. We can thus create an ebRIM extension package based on SE 1.1.[1] This might look something as follows:

Class	Description	ebRIM Registry Object	Repository Item
FeatureTypeStyle	Binding of feature type name to a styling rule.	ExtrinsicObject	FeatureTypeStyle SE encoding in XML.
	Reference to FeatureType from FeatureTypeStyle.	Association	Not applicable
OneLineResource	An online resource referenced by an xlink:href	ExtrinsicObject	Usually some XML content.
CoverageStyle	Binding of coverage	ExtrinsicObject	CoverageStyle SE

[1] There is no UML model for SE 1.1, hence the classes are only notional.

	name to a styling rule.		encoding in XML.
Rule	Defines mapping from geographic object (feature) to graphic object (symbolizer).	ExtrinsicObject	SE rule encoding in XML.
Symbolizer	Abstact class head of symbolizer hierarchy	C-Node head of a C-scheme.	Not applicable
LineSymbolizer	Linear graphical object	ExtrinsicObject	SE encoding of LineSymbolizer in XML.
PolygonSymbolizer	Area graphical object	ExtrinsicObject	SE encoding of PolygonSymbolizer in XML.
Graphic	A graphical symbol (similar to SVG graphical group)	ExtrinsicObject	SE encoding of graphic in XML.
PointSymbolizer	Point graphical object	ExtrinsicObject	SE encoding of PointSymbolizer in XML.
	Reference to a Graphic from a Point Symbolizer	Association	Not applicable
TextSymbolizer	A text graphical object.	ExtrinsicObject	SE encoding of TextSymbolizer in XML.
RasterSymbolizer	A raster graphical object.	ExtrinsicObject	SE encoding of RasterSymbolizer in XML.
Symbology Encoding Function	Transformations for categorization, formatting etc.	ExtrinsicObject	SE XML encoding of Symbology Encoding Function.

Figure 3 shows a screen shot of these objects:

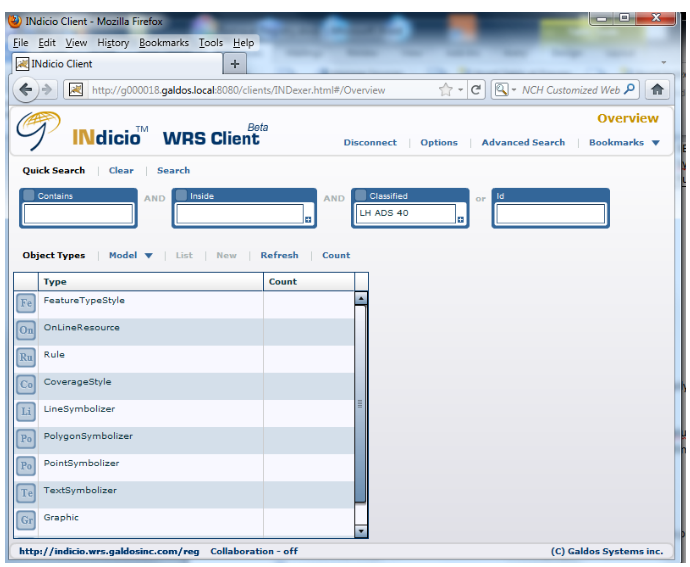


Figure 3. Registry Objects based on OGC SE 1.1

Note that the Symbolizer Classification Scheme (C-Scheme) would appear as:

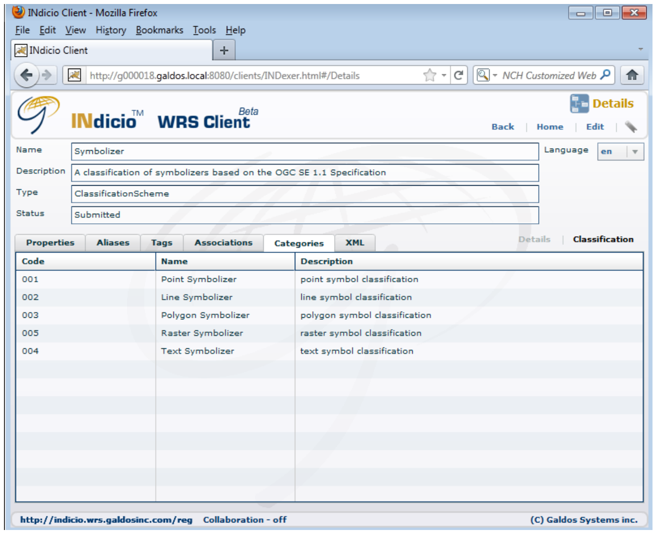


Figure 4. Symbolizer Classification Scheme

Note that there is NO Filter Registry Object. It is assumed that Filters are not easily re-used and are thus not registered objects.

Note that there is an association to a FeatureType (RegistryObject) from the FeatureTypeStyle Registry object. This is assumed to be part of another Extension Package for features defined in one or more application schema.

SLD (Styled Layer Description) documents (see OGC SLD 1.1) can also benefit from being managed through a Portrayal Registry. A possible set of RegistryObjects for an SLD is then as follows:

Class	Description	ebRIM Registry Object	Repository Item
StyledLayerDescriptor	A sequence of styled layers	RegistryPackage	Not applicable
NamedLayer	A layer accessible through a well-known name.	ExtrinsicObject	NamedLayer encoded in XML per OGC SLD 1.1
UserLayer	A custom defined layer.	ExtrinsicObject	UserLayer encoded in XML per OGC SLD 1.1

This model is shown in Figure 5.

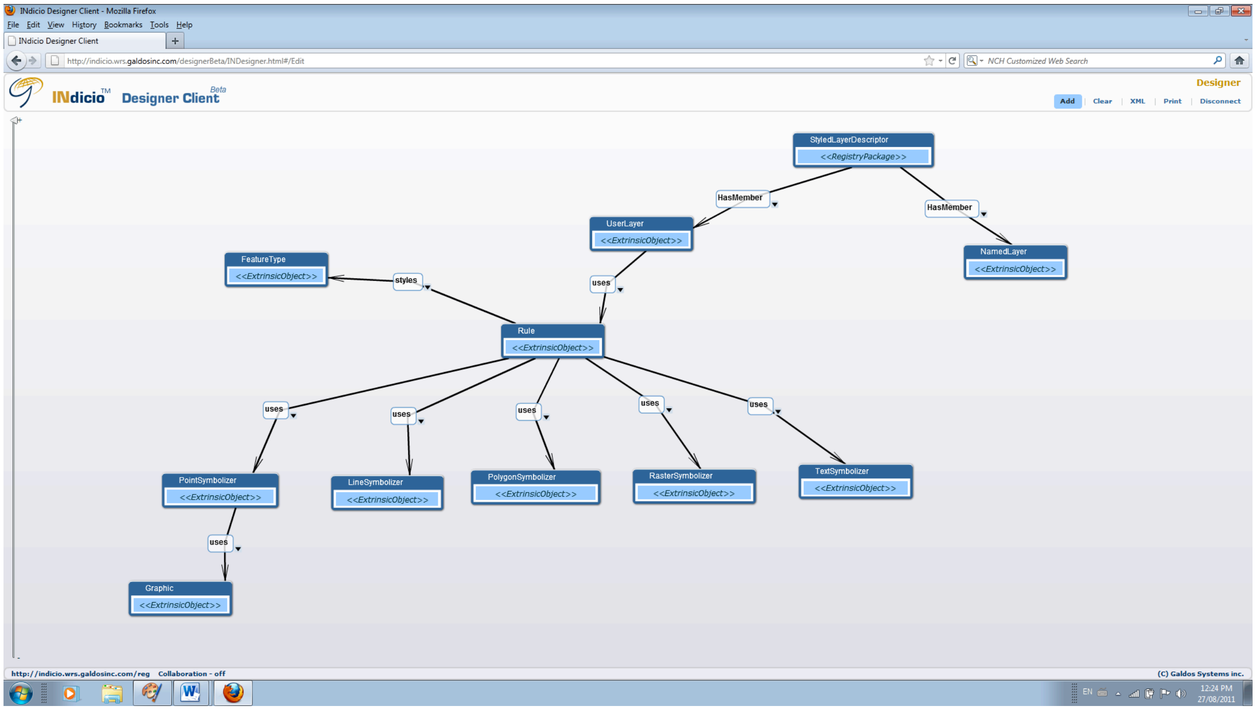


Figure 5. Portrayal Model based on OGC SE/SLD

Editor's Comment:

Should there be a single Symbol or Symbolizer ExtrinsicObject together with a Classification Scheme, or separate ExtrinsicObjects as shown in Figure 5.

4.6 Extension Package for Portrayal Registry in OWS-8

In the aviation thread (CCI) a new Extension Package for a Portrayal Registry was proposed by Carmenta based on the DGWIG Portrayal Registry[2]. A "quasi-UML" model for this registry is shown in Figure 6.

[2] Reference to DGIWG Portrayal Registry – PA5

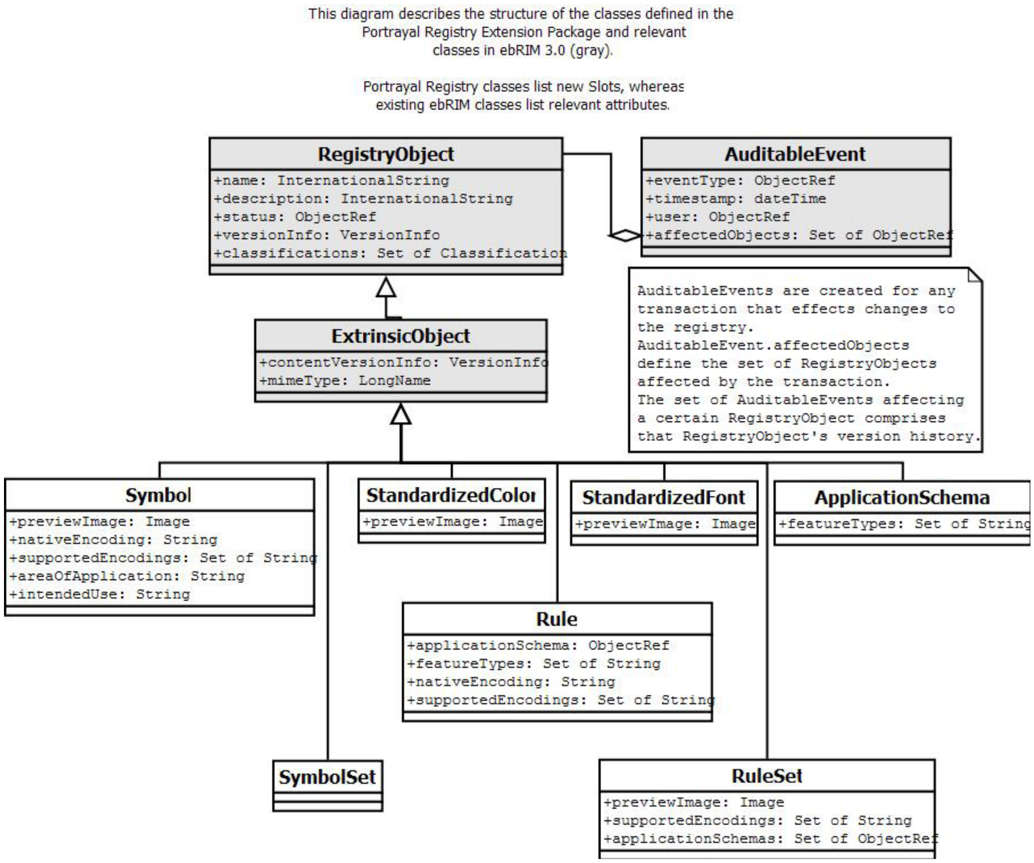


Figure 6. Quasi-UML Model for DGIWG Portrayal Registry Model

We thus have the following registry objects as implemented by Carmenta.

Class	Description	ebRIM Registry Object	Repository Item
Symbol	A graphical object used by a rule.	ExtrinsicObject	Encoding of the symbol
SymbolSet	A related group of symbols.	ExtrsinicObject	?
Rule	Bind feature types to symbols. Maps geographic object (feature) to graphic object (symbol)	ExtrinsicObject	Encoding of the rule
RuleSet	A related group of rules.	ExtrinsicObject	?
StandardizedColor	Named colours to be used by symbols.	ExtrinsicObject	?
StandardizedFont	Named fonts to be used by symbols.	ExtrinsicObject	?
ApplicationSchema	An application schema as may be defined through a feature catalogue.	ExtrinsicObject	The schema encoded in XML schema?
UsesSymbol	Reference to a Symbol from a Rule	Association	Not applicable

Editor's Comment:

1. Should consider RegistryPackage for SymbolSet, and RuleSet.
2. Should consider Classification Scheme for Standardized Color(s). Note that the usual way to handle an associated image is to use a Browse Graphic. This is a separate ExtrinsicObject that holds the image that would then be associated to the C-Node.
3. Should consider Classification Scheme for Standardized Font(s). Note that the usual way to handle an associated image is to use a Browse Graphic. This is a separate ExtrinsicObject that holds the image that would then be associated to the C-Node.
4. Should consider RegistryPackage for ApplicationSchema, i.e. a collection of Feature types.

One issue that arose in the thread discussions was whether or not there should be a separate registry for application schemas and how fine grained that registry should be. The scheme outlined above uses an association to relate the Rule and RuleSet to the application schema.

Editor's Comment: Since a Rule is a member of a RuleSet it would seem that an association from the Rule to the Association Schema alone would be sufficient.

A number of other associations are part of the model as shown in Figure 7.

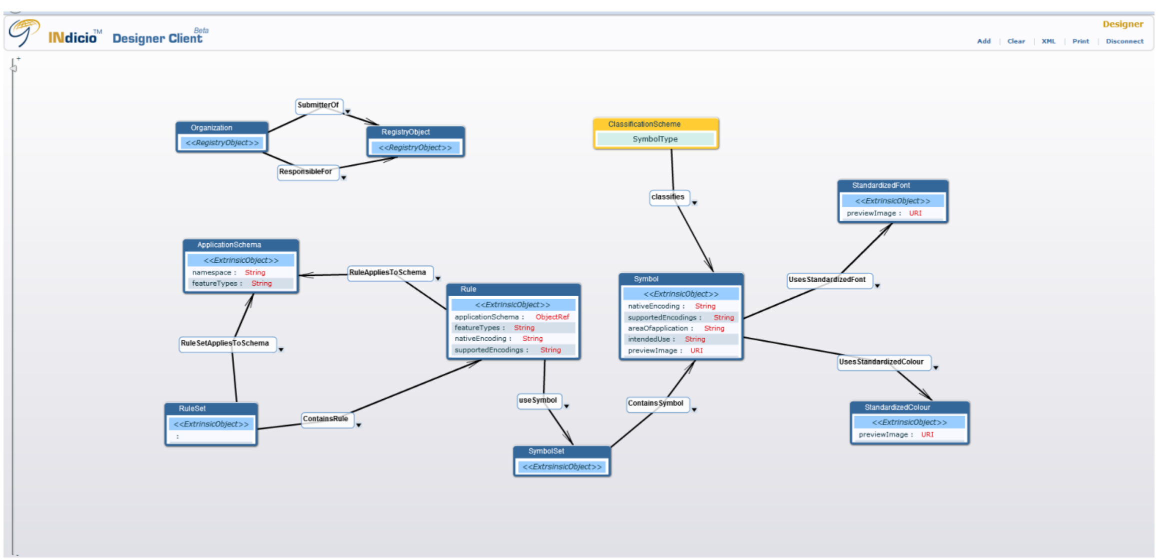


Figure 7. ebRIM Extension Package for Portrayal

Editor's Comment:

An association (submittedBy) is used to support governance of the Portrayal Registry objects. It is suggested that this be done using life cycle metadata (e.g. Submitted, Approved, etc), in conjunction with the ebRIM audit trail.

4.7 Discussion

Three different extension packages have been presented in this report. A number of characteristics of these extension packages are of interest:

1. Use of ExtrinsicObjects for the various portrayal registry concepts, with the detailed encoding captured in an associated Repository Item. (All) This approach allows different encodings to be used for symbols and styles.

2. Use of an "encoding" slot that says how the corresponding Repository item is encoded. (DGIWG model).

3. Use of classification scheme for different types of symbols. Used in both OWS-7 and DGIWG. This would seem to be the better approach also for OGC SLD/SE based model.

4. Means of associating features and graphic objects.

   a. In the OWS-7 model this was handled by the abstract DataProduct Portrayal.

   b. In SE/SLD this was handled by the Rule object with an explicit association "style" referencing a Feature ExstrinsicObject, and an association "uses", referencing one or more of the Symbolizer types.

   c. In DGIWG this was handled by both Rule and RuleSet objects with explicit associations to an ApplicationSchema ExtrinsicObject. While this association makes sense for the case of a RuleSet, it might be better to have an association from the Rule to the Feature type that the rule is applied to.

5. All three models provide a concept of grouping styling rules. This is the Styled Layer Descriptor (SLD) in the SLD/SE model, the Rule Set in the DGIWG model, and the DataProductPortrayal with an SLD document as the Repository Item (RI) in the OWS-7 model.

   The term StyledLayerDescriptor has a somewhat mixed heritage. It was originally proposed as a means of describing a visual (map) layer generated by a Feature Portrayal Service, and later became a means of describing so called named styles and layers as returned by any Web Map Service. SLD v1.1 harmonized these two different approaches to the description of style in the context of map layers.

   This then raises the question as to the name that should be used in the PortrayalRegistry for a group of rules constituting a style. One choice is of course SLD, but that seems overly bound to the particular encoding as specified in SLD 1.1. An alternative, as proposed in the DGIWG model is that of rule set. A third choice is that of DataProductPortrayal from OWS-7. Perhaps a fourth choice, combining elements of both, might be StyleRuleSet, or StyleRules, or simply Style.

6. The DGIWG model uses the name SymbolSet for a collection of related Symbols (e.g. S-52 Symbols), OWS-7 used Symbol Library, and the concept was not part

of the SLD/SE model. The concept is clearly both useful and widely used. Likely Symbol Library is more commonly used term.

A synthesis of the models discussed above is presented in Figure 4.7-1

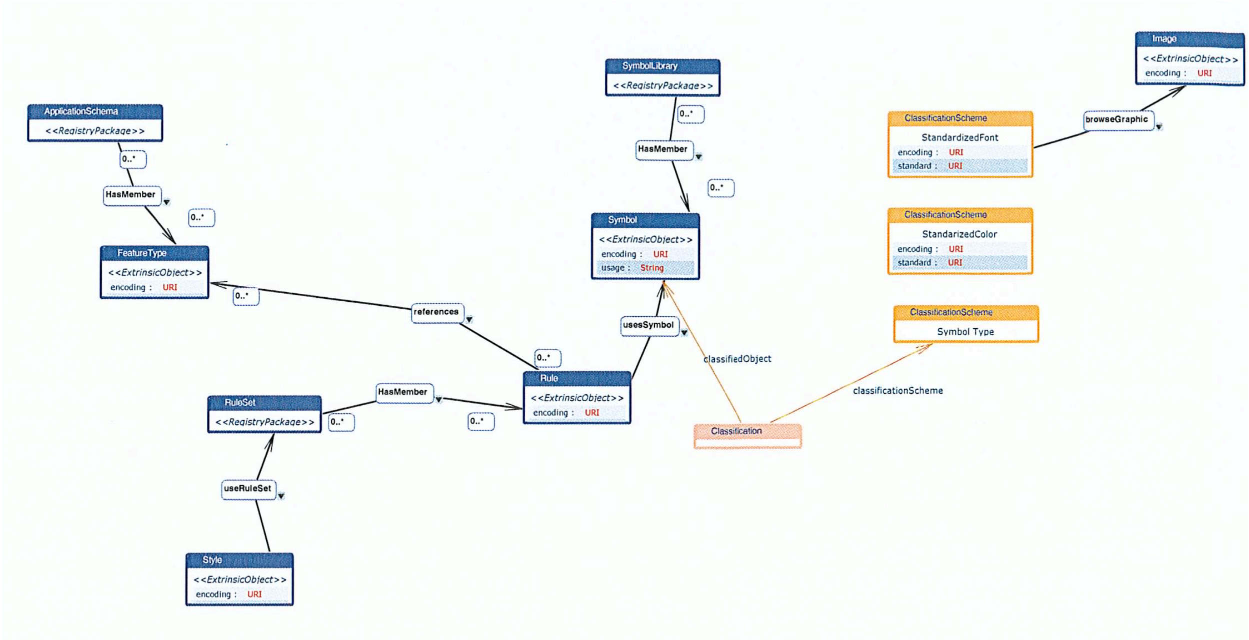


Figure 4.7-1 Proposed Registry Model - Synthesis

4.8 What questions should the Portrayal Registry Address?

One approach to creating the Portrayal Registry extension package is to specify the questions that such a registry should be able to answer, and then analyze these questions to develop the model.

The following questions are believed to be representative of a broad set of portrayal requirements:

1. Find and retrieve all symbols for the portrayal of features with point geometries.

2. Find and retrieve all styling rules that can be applied to a particular feature type.

3. Find and retrieve a style (set of styling rules) applicable to a particular schema (set of feature types).

4. Retrieve a graphic that can be used to symbolize a feature with point geometry.

5. Find and retrieve all rules that could be used with a feature with specific properties. Note that this requires knowledge of the properties of the feature type. Note also that the rule may only apply to a subset of the rules.

6. Symbols may be parameterized. Find all symbols with a specific set of parameters. Find the parameters of a selected symbol (e.g. Circle: Diameter, Colour).

Note that we exclude questions that might be asked of a more general registry model such as 1) Find all WMS with maps for a given region or 2) Find all WFS that can provide certain feature types for a given area , 3) Find a Feature Portrayal Service that can process a specific rule set (style, SLD, StyleRuleSet).

In utilizing a Feature Portrayal Service, we would want to locate a WFS, find the schema from the WFS (GetCapabilities, DescribeFeature), and then find a suitable style (see item 3 above in Section 4.8) that can be applied to the WFS. Another use case would be to get an application schema from a WFS, or Registry (not necessarily focused on styles), and then retrieve a symbol library (see 4.8, items 1, 5, and 6) and construct a set of styling rules. Register the styling rules (rule set) in a Portrayal Registry.

Another user might then find this style and use a different FPS to style the features from the WFS and create a map.

These use cases are summarized in Figure 8.

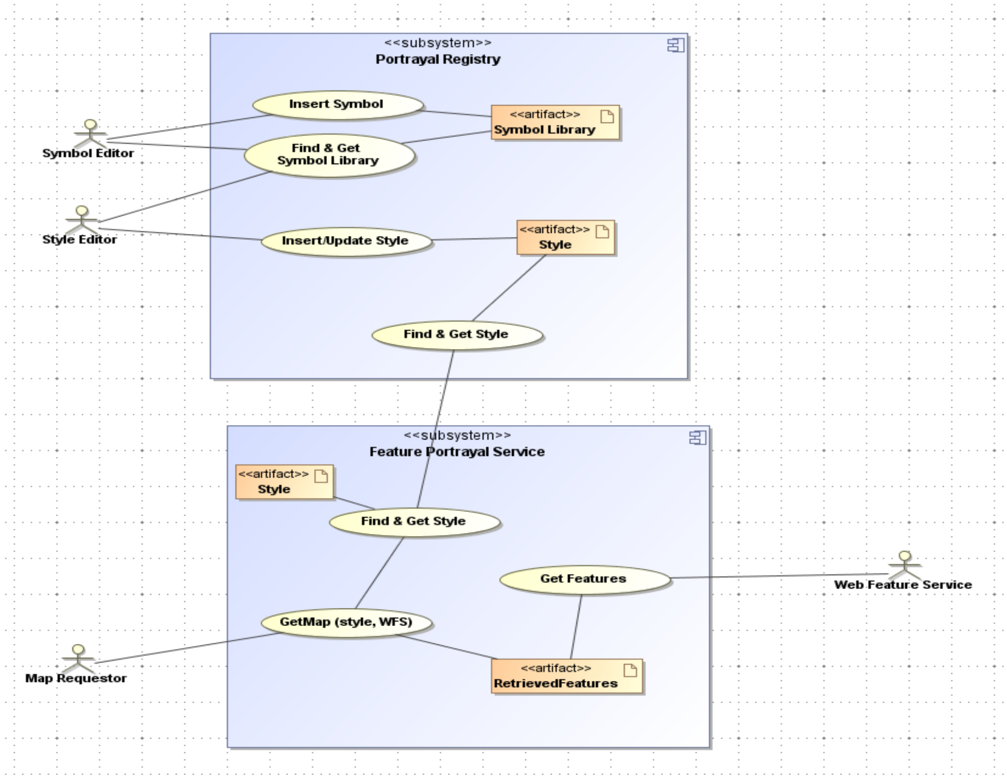


Figure 8. Portrayal Registry Use Cases

From these questions we can see that:

1. We need a symbol library construct in the Registry.

2. We need a Rule construct and we should be able to locate Rules based on the features and the symbols that they apply to. Is symbol the best choice of word? SE uses the "anthropomorphic" Symbolizer. Is this better than symbol? Neither seems entirely correct since in the case of photo-realistic styling, we would not think of the graphic output as a symbolic representation of the geography. Would graphic object be better?

   Note that in some styling cases, the style rules effectively express symbol properties as functions of feature properties. In this case, discovery via the Registry must enable us to find the rules which refer to features with specific properties. Similarly, we would like to be able to find Symbols with specific properties. Should this be a Registry task or something that must be handled by the style editor or map request client.

4.9 Recommendations

The following recommendations are based on the combined experience of developing ebRIM extension packages for Portrayal from OWS-7, OWS-8 Aviation and the OGC Special Activity Airspace (SAA) Pilot project.

1. ExtrinsicObjects should be used to capture the key concepts that would be used in the Portrayal Process, but their specific encoding should be in the Repository Item, with a slot to specify the encoding. This is done in the DGIWG portrayal registry.

2. Collections such as Symbol Set, Rule Set etc. should be modeled as Registry Packages, using a standard "HasMember" association.

3. StandardColors, StandardFonts are a good idea. We recommend that these be handled as Classification Schemes. The Classification Nodes can have properties such as RGB value etc. Additionally we could have a color image ExtrinsicObject classified by the StandardColors C-Scheme. The same approach can be applied to StandardizedFonts. We recommend these be named StandardColors and StandardFonts respectively.

The recommended model is summarized in the following table:

Class	Description	ebRIM Registry Object	Repository Item
Image	An image used as a browse graphic	ExtrinsicObject	Bit map encoding of the image.
Symbol	A graphical object used by a rule.	ExtrinsicObject	Encoding of the symbol as SE or otherwise.
Symbol Library	A related group of symbols.	Registry Package	Not applicable
Rule	Bind feature types to symbols. Maps geographic object (feature) to graphic object (symbol)	ExtrinsicObject	Encoding of the rule
RuleSet	A related group of rules.	Registry Package	Not applicable
Style	A document defining the appearance of a map.	ExtrinsicObject	OGC SLD or other encoding.
useRuleSet	Reference from Style to a RuleSet	Association	Not applicable
StandardizedColor	Named colours to be used by symbols.	Classification Scheme with association to a browse graphic as ExtrinsicObject.	Color preview bit map. (optional)
StandardizedFont	Named fonts to be used by symbols.	Classification Scheme with association to a browse graphic as ExtrinsicObject.	Font preview bit map (optional)
FeatureType	Feature type name with associations to type definition, properties. Namespace,version as slots.	ExtrinsicObject	GML encoding of feature (GML element declaration)
ApplicationSchema	An application schema as may be defined through a feature catalogue.	Registry Package	Not applicable.
usesSymbol	Reference to a Symbol from a Rule	Association	Not applicable
styleBy	Reference to a Style	Association	Not applicable
browseGraphic	Reference to a browse	Association	Not applicable

	graphic		

This combines features of all of the models presented in Section 4.

5 Feature Portrayal Service –Portrayal Registry Interaction

5.1 Overview

A Feature Portrayal Service is a form of Web Map Service (WMS) that obtains feature data (e.g. GML) from one or more Web Feature Services, and applies styling rules to create the map that is returned to a Map Request Client. This basic concept is illustrated in Figure 9.

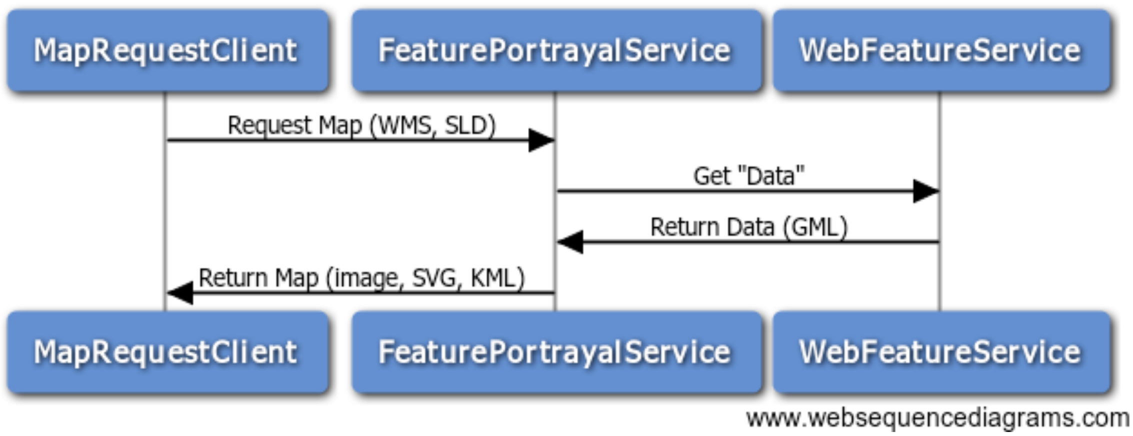


Figure 9. Concept of a Feature Portrayal Service

Note that in this diagram the actual styling rules are supplied by the requesting client and there is no Portrayal Registry (See 4.2.2).

An important issue in connection with an FPS, is the applicability of the styling rules (SLD) with respect to the referenced WFS. The styling rules are in effect defined relative to a GML application schema (call this a reference schema), and any WFS referenced in the requests from the FPS must be consistent (i.e. have the same feature type names and at least the same property names and types that are referenced in the styling rules) with the reference schema.

5.2 Portrayal Registry

A Portrayal Registry can enhance the picture of 9 by managing the styling rules (SLD) and symbols (SE), providing an association to the reference schema(s), and thus offloading these tasks from the client. This immediately facilitates the standardization of styles and symbols and hence their sharing across a specific application community. This is illustrated in Figure 10.

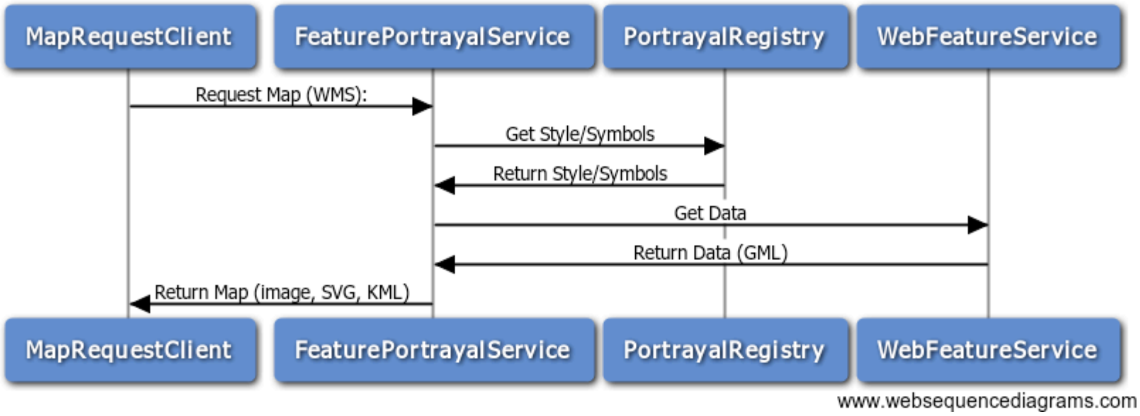


Figure 10. Concept of a Portrayal Registry

While Figure 10 illustrates the general concept of a Portrayal Registry, there are other potential modes of interaction, depending on the capabilities of the FPS. These are discussed in the following sections. For further details on the "standard" case of an FPS and Portrayal Registry see the OWS-7 Engineering Report entitled: OGC® OWS-7 Engineering Report - Aviation Portrayal.

Note that in Figure 10 the styling service is an OGC FPS that only "understands" styling documents expressed as OGC SLD documents with SE encodings for feature/coverage styles etc. In Section 4 however, we examined the case of a Portrayal Registry which supported other encodings than SLD/SE. In such a case it will be necessary to transform the Portrayal Registry response into an SLD/SE so that it can be consumed by the FPS. This use case was developed in some detail in the OWS-8 CCI Portrayal Registry TIE and is discussed in the following sections.

We thus have the following roles for the Portrayal Registry:

1. Manage information about available styling services (i.e. FPS) so that they can be discovered and used for portrayal.

2. Manage information about available Web Feature Services (WFS) and Web Map Services (WMS) so that they can be discovered and provide data (to be portrayed) or background maps that participate in the portrayal. While clearly this information is required for Portrayal, it should be considered for registration as more general information which is of importance for portrayal.

3. Manage portrayal styling rules which are applicable to the styling of particular feature types.

4. Manage symbols and symbol libraries in support of the styling rules in 3.

5. Manage the GML application schemas that are "referenced" by the styling (portrayal) rules. While clearly this information is required for Portrayal, it should be considered for registration as more general information which is of importance for portrayal.

5.3 Portrayal Registry Service (PRS) and Feature Portrayal Service (FPS) Interaction

The sections below describe the three principal ways we see for a client to interact with the portrayal registry in conjunction with an FPS.

5.3.1 Legacy (SLD 1.0) FPS

The first use case describes a situation where a legacy FPS, i.e. an FPS not capable of communication with the PRS, is used by the client. This could for example be an FPS which only supports version 1.0 of the SLD standard, which does not contain the OnlineResource elements.

The client (MapRequestClient) is then responsible for communication with the PRS, creating the SLD document and delivering it to the FPS (see Figure 11).

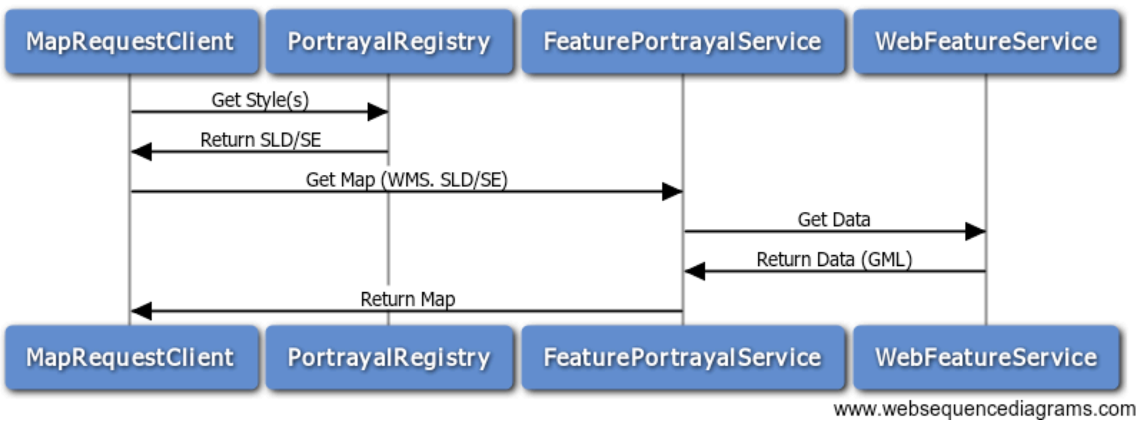


Figure 11. Legacy FPS – Portrayal Registry Interaction

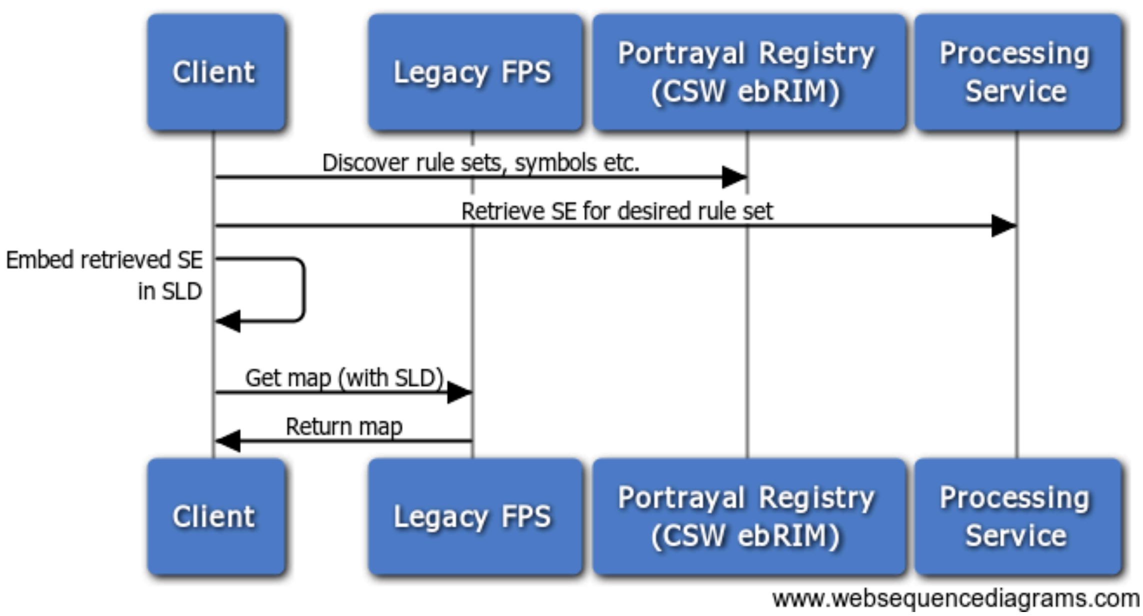
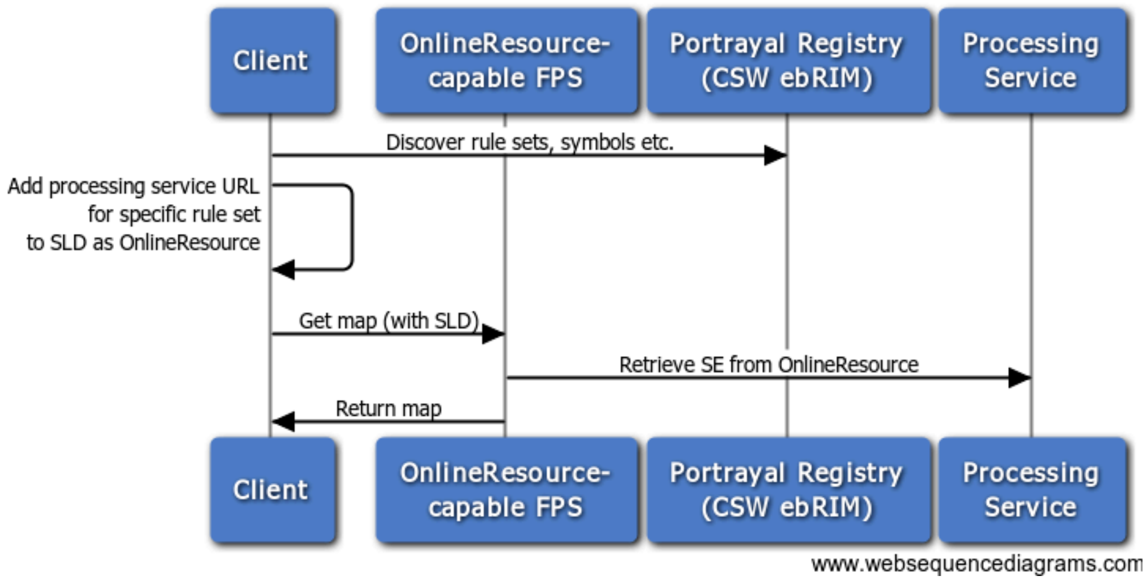


Figure 12. PRS Interaction with FPS via Processing Service

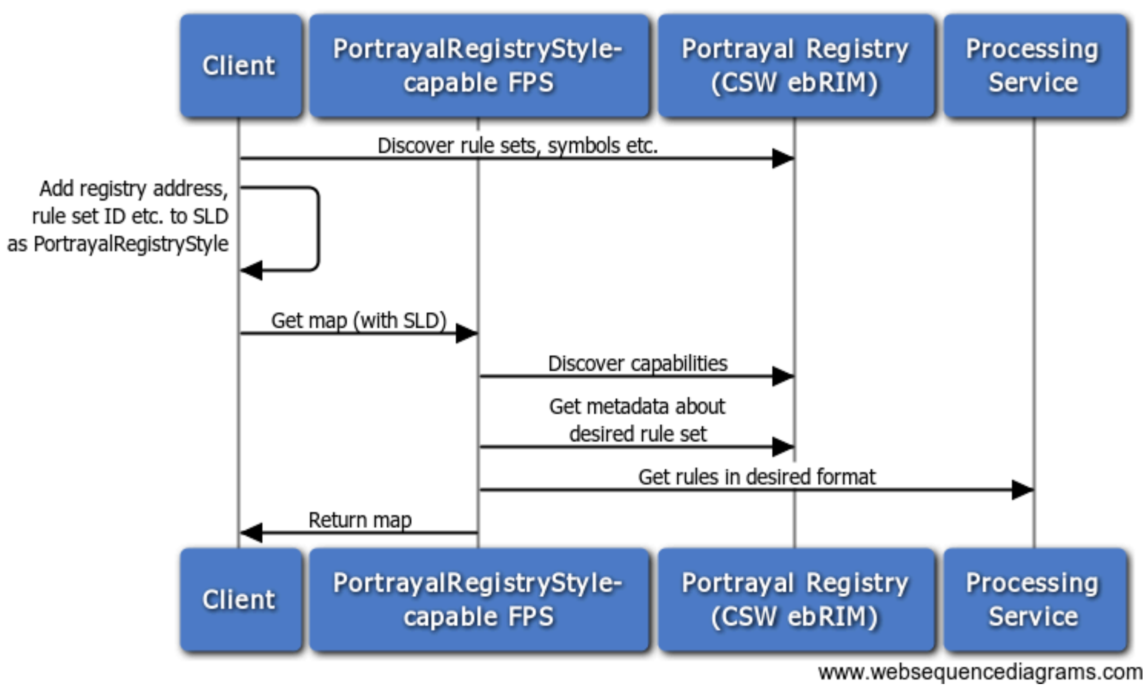
5.3.1.1 SLD/SE 1.1 FPS with OnlineResource Support

The second use case describes a situation where the FPS is capable of interpreting OnlineResource references, removing the responsibility from the client to construct a 'vanilla' SLD document.



5.3.1.2 FPS with PortrayalRegistryStyle Support

The third use case describes a situation where the FPS is capable of interpreting the proposed PortrayalRegistryStyle elements.



DGIWG views the OWS-8 testbed as a way to evaluate the usefulness of a PortrayalRegistryStyle element. Compared to directly linking to an SE 1.1 document (via an OnlineResource reference), there are several potential benefits in this approach, including:

- It allows the FPS to check in which formats the portrayal rules can be returned, and pick one that it supports. For example, an ESRI server talking to an ESRI portrayal registry could conceivably opt to retrieve the rules as some sort of "MXD fragments".

- If there was a way to determine metadata like the application schema and feature class and from a WFS server, it could potentially allow the FPS to automatically only retrieve rules for that specific schema and class from the registry, thereby removing the need for the client to specify this information as part of the PortrayalRegistryStyle element.

- If the FPS detects - based on the metadata it retrieves from the registry - that the portrayal rules correspond to a symbol standard it has support for (e.g. built-in rendering support for IHO S-52), it may elect to use its own styling engine, rather than the SE-encoded rules, to do the portrayal. Note that the author of the OWS-7 Aviation Portrayal ER (Roger Brackin) identified the need for such a mechanism.

5.4 Evaluation of support for LTDS Portrayal Standard

5.5 KML Styling Support

A number of objectives were outlined for feature styling, expressed via KML. These are summarized in the following table:

Nr.	Requirements	Plans
4.4.2.2.4	Evaluate and demonstrate if the same style resources could be used for OGC Portrayal Services and KML data, i.e. KML and SLD/SE would simply be different representations of the same resource.	SE symbolizers associated with each rule would be provided by the portrayal registry in two representations, the SE representation and the KML representation (kml:Style/kml:StyleMap). Each style/style map would be referenced using a persistent URL that can be embedded into KML.
4.4.2.2.5	Evaluate the use of portrayal registries within KML. KML features reference styles, while OGC Portrayal Service requests reference SLD (containing a set of layers with one or more styles). These resources would need to be managed consistently in the registry.	The experiences with the use of a portrayal registry to support addressing the requirements 4.4.2.2.4 will be documented in a ER from the perspective of the KML implementation in 4.4.2.3.2.
4.4.2.3.1	Add KML support for child elements in a BalloonStyle to specify, for example, styles and scripts.	Out of scope for OWS-8
4.4.2.3.2	Advance KML encoding support for different styles per feature type. Currently within KML all features of a particular feature type are rendered using the same	Since this has to be addressed in the encoding rule, i.e. in the ShapeChange transformation, we still need the tagged value in the UML model to represent the styling information. However, instead of a fixed URL for the KML feature type

	style.	style, one or more URLs would reference the style information in the portrayal registry. These styles in the portrayal registry would then be accessed by ShapeChange in the step of creating the XSLT stylesheet for the GML-to-KML conversion and the Filter expression and scale information would be used to generate the XSLT. Note that the complexity of the supported filter expressions must be simple (only a subset of Filter will be supported - filter expressions will be limited to support selection based on comparison of attribute values of the feature, e.g. the building function, with literal values) and restricted to feature properties embedded inline in the GML encoding. The scale information must be passed as a parameter to the stylesheet. In the encoding of the KML (i.e. the XSLT conversion from GML), the appropriate KML style element referenced for each feature instance would be determined based on the selection criteria formulated using Filter Encoding. The approach will be implemented in ShapeChange (i.e., it could also be seen as part of CciShapeChangeExtensions) and tested with the NGA test data. The necessary styles are expected to exist in the portrayal registry. This requires that the approach to requirement 4.4.2.2.4 is implemented as described above
4.4.2.3.3	Advance KML encoding to support layers. KML does not have a concept of layers and styles associated with these layers. Instead, each feature references the style in which it should be rendered.	We do not consider it necessary that a notion of layers is introduced in KML. The solution proposed for requirement 4.4.2.3.2 should be sufficient for the functional requirements stated in the OWS-8 RFQ. I.e., out of scope for OWS-8.

5.5.1 KML as a feature encoding

Some participants in OWS-8 argued that KML can be considered as a feature encoding and one that is compliant with ISO 19109. While there is no possibility to create a named type of object, (e.g. road) in KML, it is possible to attach attributes to a KML Placemark using ExtendedData, and some argue that this is sufficient to make a Placemark a feature, if only a kind of generic one. Note that the original intention of the KML ExtendedData element was to provide data values to parameterize the balloon style of the Placemark – for example the width of a placemark LineString can be drawn in proportion to a data value such as ‘numberOfLanes’ and the balloon style can reference such data using the following reference encoding:

```
$[numberOfLanes]
```


However, despite the intention for balloon styling, the ExtendedData usage has no such schema-enforceable restriction in the OGC KML standard.

Following this line of argument, KML "features" are then served by a WFS and styled by an FPS for presentation as KML. This would mean that Placemarks from the WFS are modified by the FPS by the addition of the KML <Style> elements.

5.5.2 KML as a symbol encoding

From a symbol encoding viewpoint, KML can be understood as a graphical language like SVG (W3C Scalable Vector Graphics), in particular in view of the proposed SVG Map specification. In this case, the KML Placemark can be seen as analogous to a Symbolizer in the OGC SE v1.1 specification. Features, expressed in another language (typically GML) are then styled by an FPS to KML according to the styling rules and applied filters. In this view, Style in KML is not a FeatureTypeStyle in the sense of OGC SE v1.1. In OGC KML 2.2, there are several types of styles (<Style> element children), namely <IconStyle>, <LabelStyle>, <ListStyle>, <LineStyle>, and <PolyStyle>, and each style element defines colour, thickness and other elements of the associated presentation element. KML Placemarks would be generated by the FPS, including the Style information.

With respect to the Portrayal Registry, we would anticipate, for example, that a particular Symbol ExtrinsicObject (classified as a PolygonSymbol, and with encoding KML v2.2) would contain a KML <PolyStyle>. The following table summarizes the anticipated KML content for the different Symbol types:

Symbol Type (PR) (EO)	ExtrinsicObject Repository Item	Comments
Point	<IconStyle>	
Line	<LineStyle>	
Polygon	<PolyStyle>	
Text	<LabelStyle>	

Note that OGC KML 2.2 styling supports constructs which are not available in SE v1.1. In particular:

1. The ability to incorporate attribute values via <ExtendedData>.
2. The <ListStyle> which applies to the appearance of items in a "legend".

The following table provides a rough comparison of SE v1.1 Symbolizers and the corresponding KML Style elements.

KML[3] Style Element	SE Symbolizer Element	Comments

[3] Only KML 2.2 elements are used, i.e. no elements from extension namespaces are included.

<IconStyle> <Icon> <heading> <scale> <color> <hotspot> <hotspot>	<PointSymbolizer> <Graphic> <Rotation> <Size> <Opacity> <ColorReplacement> <Displacement> <AnchorPoint>	 Both may reference "graphic" via URL. Both in degrees. SE allows negative values. SE Size is absolute size in specified units (uom). <scale> KML is a relative scaling based on default size. Not available in KML Can override colors of the icon or graphic.
<LineStyle> <width> <color>	<LineSymbolizer> <Stroke> <GraphicFill> <GraphicStroke> <SVGParameter> <SVGParameter> <PerpendicularOffset>	 Not available in KML[4] Not available in KML stroke-width (SVG parameter) stroke (SVG parameter)[5] Not available in KML
<PolyStyle> <fill> <outline>	<PolygonSymbolizer> <Fill> <GraphicFill> <SVGParameter> <Stroke>	 fill in KML is just a Boolean value Not available in KML Controls colour and opacity. KML color controls polygon fill color. For SE see <Stroke> under <LineSymbolizer> In KML <outline> is just a Boolean value. Appearance is determined by current <LineStyle> - similar to <Stroke> in SE.
<LabelStyle> <color> <scale>	<TextSymbolizer> <Fill> <SVGParameter> <Font> <LabelPlacement> <Halo>	KML color as discussed above. As in discussion of Fill above for SE. <scale> KML controls text size. SVG Parameters are used to control font, size etc. in SE. Not available in KML Not available in KML

5.6 Transforming Styles

Since the Portrayal Registry can store more than one type of style encoding (via the encoding Slot), it will be, in many cases, necessary to transform a retrieved style to a different encoding, before it can be used by a styling service. If the styling service is, for example, an OGC Feature Portrayal Service (FPS), it can only make use of style documents that are encoded as a Styled Layer Descriptor.

This raises two questions, both of which were investigated to some degree in the TIE, namely:

- Where should the transformation be done.

[4] Means there is no comparable capability in KML

[5] SVG Parameters (like real SVG) provide many more controls such as opacity, dash array etc than possible in KML.

- How general can such transformation be.

5.6.1 Where should the transformation take place

Two choices were proposed in the project, namely:

1. Use an external Web Processing service.
2. Require that the transformation be performed by the Registry.

5.6.1.1 Using a Web Processing Service

A Web Processing Service is a general purpose web service that can perform any advertised operations on data supplied to the service. In this case the input to the service would be styling information obtained from the Portrayal Registry, and the output would be an SLD that could be consumed by the FPS.

The advantages of this approach are:

1. The WPS is easily discovered from a registry/catalogue service.
2. There is no need to synch static copies of a FeatureTypeStyle which have simply different encodings.

The disadvantages of this approach are:

1. Either an intervening client must connect the PR and the WPS, or there be some form of orchestration of the services, or (worst case), the WPS must know about the PR and the WPS. The client based scenario might be considered in the case, where a user knows about an FPS, gets the styling information from the PR, and then finds an appropriate WPS to do the transformations.
2. The WPS being a completely separate service must be passed all context information required to execute the transformation (WPS implementation and service chaining overhead for a single transformation).
3. Performance can be expected to be degraded since several web service interactions are required to generate the presentation output.

5.6.1.2 Registry Utility

Many Registry Services provide a transformation utility in support of the CSW-ebRIM "view" parameter. While normally restricted to "full", "brief" etc views of registry content, this could be extended to support the return of results in different forms.

A particular transformation is then invoked depending on the value of the "view" parameter selected.

The advantages of this approach are:

1. A request mechanism already exists which can be used by a registry (user) client or by an FPS ("give me the results in this form") to get the style in the desired form.
2. The PR has the complete context for executing the required transformation.

The disadvantages of this approach are:

1. The 'view' parameter approach has not (yet) been standardized by the OGC CSW-ebRIM standard. However, this is likely to happen as three of the OGC CSW-ebRIM vendors (ERDAS, CubeWerx, Galdos) have implemented such an enhancement.

6

6.1 Symbol Libraries and Style Construction

6.2 Overview

One use case not considered in the TIE, is the creation of map styles using some form of map style editor. This is illustrated in the following sequence diagram.

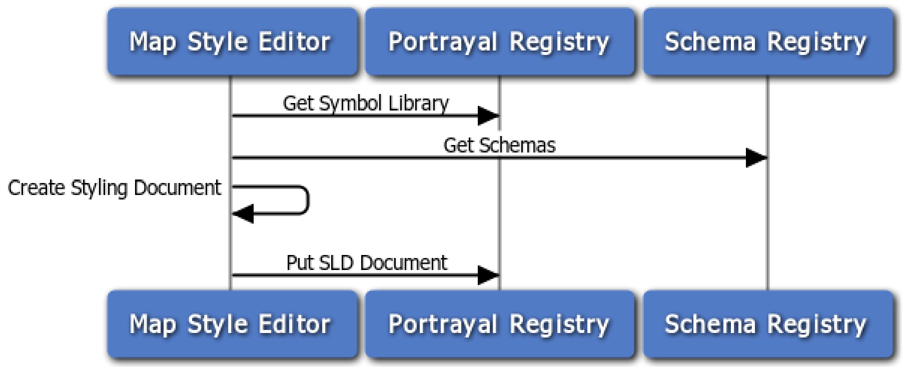


Figure 6.1-1 Map Style Editor Creation

As shown in the sequence diagram, the user responsible for a map style, loads a selected set of feature types (logical schema), and a symbol library (e.g. Point, Line and Polygon Symbols) and then proceeds to construct style mappings (e.g. SLD) using the Map Style Editor application. The Editor then registers the style mapping (style) in the Portrayal Registry.

6.3 Symbol Libraries

Symbol Specifications
OWS-8 Cross Community Interoperability Tread

The symbols are defined in the National System for Geospatial Intelligence (NSG) Portrayal Standard for Local Topographic Data Store (LTDS) Data, Draft 30 September 2010, except as noted below.

Road Curve - 2,302 instances

AP030 Road Curve 1
MES=1001 (True) and LTN>=2
Symbol ID:
LN_Road_AllWeather_HardSurface_Operational_Median_MultipleLanes

AP030 Road Curve 2
MES=1000 (False) and LTN>=2
Symbol ID:
LN_Road_AllWeather_HardSurface_Operational_NoMedian_MultipleLanes

AP030 Road Curve 3
LTN=1
Symbol ID: LN_Road_AllWeather_HardSurface_Operational_NoMedian_1Lane

Building Surface - 2,428 instances

AL013 Building Surface 1
HGT<46 meters
Symbol ID: AR_Black_Building

AL013 Building Surface 2
HGT>=46 meters
Collapse to point
Symbol ID: PT_Blue072_VO

Building Point - 10,349 instances

AL013 Building Point 1
HGT<46 meters
Symbol ID: PT_Black_Building_Landmark

AL015 Building Point 2
HGT>=46 meters
Symbol ID: PT_Blue072_VO

Built Up Area - 162 instances

AL020 Built Up Area Surface 1
BAC=2 (Dense)
Symbol ID: AR_Black-54_BUA_Dense
Being changed to red-brown, 42% which is R=212, G=148, B=148

AL020 Built Up Area Surface 2
BAC=1 (Sparse) or 3 (Moderate)
Symbol ID: AR_Black-31_BUA_Sparse
Being changed to red-brown, 12% which is R=243, G=224, B=224

Tidal Water - 3 instances

BA040 Tidal Water Surface
Symbol ID: AR_Cyan-31_TidalWater

Railway - 91 instances

AN010 Railway Curve
Symbol ID: LN_Black_Railway_SingleNormalGauge_Non-Electric

River Surface - 14 instances

BH140 River Surface 1
HYP=1 (Perennial) or 999999 (no information)
Symbol ID: AR_Cyan_River_Perennial

BH140 River Surface 2
HYP=2 (intermittent) or 4 (dry)
Symbol ID: AR_Cyan_River_Non-perennial

River Curve - 197 instances

BH140 River Curve 1
HYP=1 (Perennial) or 999999 (no information)
Symbol ID: LN_Cyan_River_Perennial

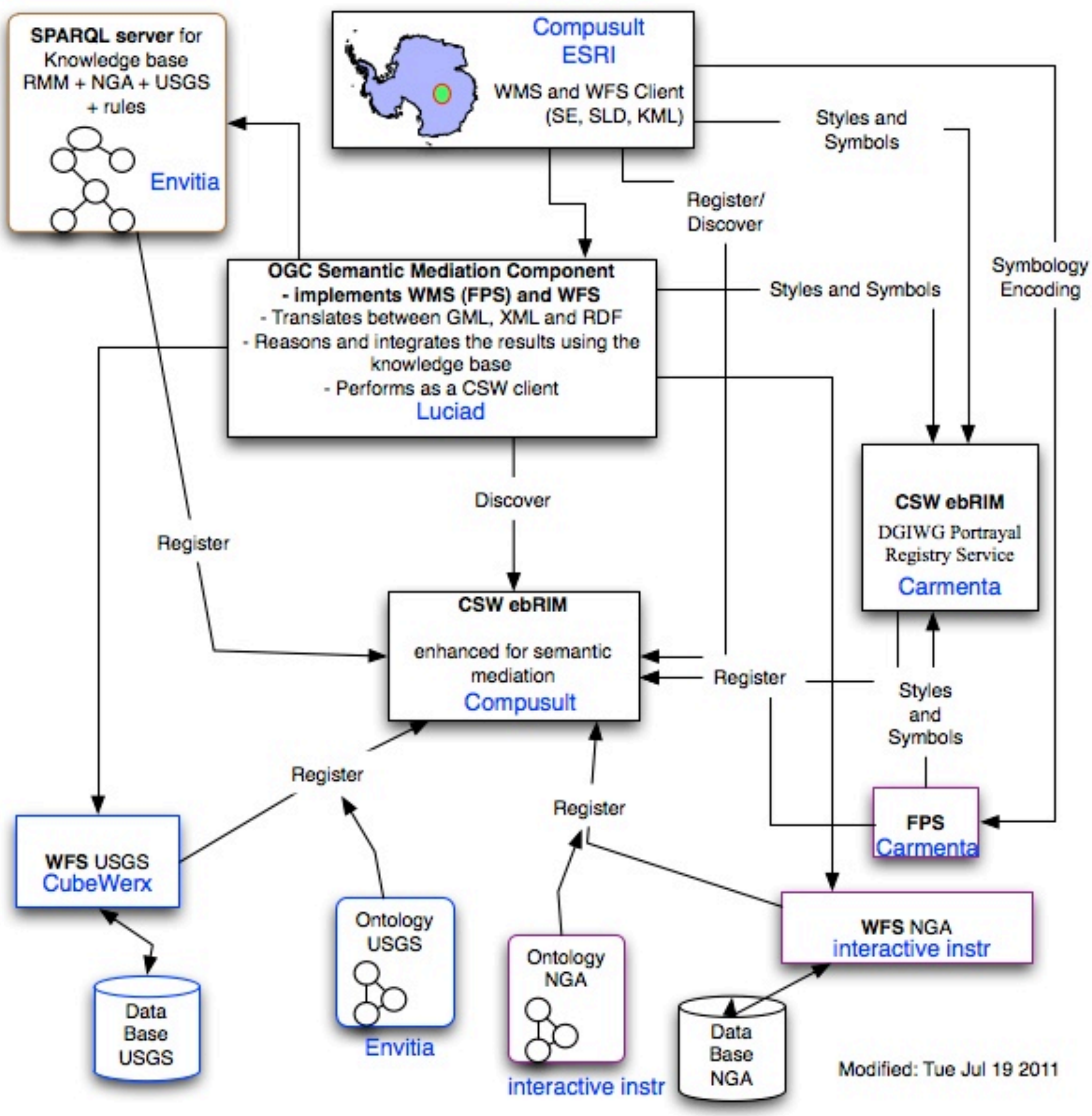
BH140 River Curve 2
HYP=2 (intermittent) or 4 (dry)
Symbol ID: LN_Cyan_River_Non-perennial

Forest Surface - 81 instances

EC015 Forest Surface
Symbol ID: AR_Green362-31_Woodland_Unknown

7 Implementations

7.1 Overall architecture of the CCI Thread



7.2 Registry (CSW-ebRIM)

Three CSW-ebRIM registries were deployed in the TIE:

1. Carmenta (Primary Portrayal Registry).
2. Compusult (General Registry).
3. CubeWerx (Portrayal Registry)

7.2.1 Proposed (DGIWG) Registry extension package for Portrayal

```
<?xml version="1.0" encoding="utf-8"?>
<!--
        Portrayal Registry Extension Package for CSW-ebRIM.
        Last updated: 2011-05-17
-->
```



```
<rim:RegistryPackage xmlns:rim="urn:oasis:names:tc:ebxml-regrep:xsd:rim:3.0"
  id="urn:x-ogc:specification:csw-ebrim:package:PortrayalRegistry"
  objectType="urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject:RegistryPackage">

		<rim:Name>
			<rim:LocalizedString xml:lang="en-US" charset="UTF-8"
								value="Portrayal Registry
extension package for CSW-ebRIM" />
		</rim:Name>
		<rim:Description>
			<rim:LocalizedString xml:lang="en-US" charset="UTF-8"
								value="Provides Portrayal
Registry extensions to the Basic package of the CSW-ebRIM catalogue profile."/>
		</rim:Description>

		<rim:RegistryObjectList>

			<!-- Extensions to canonical ObjectType scheme
			 (new ExtrinsicObject subclasses defined in this extension package) -->
			<rim:ClassificationNode code="RuleSet"
			  objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ClassificationNode"
			  lid="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:RuleSet"
			  id="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:RuleSet"
			  parent="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ExtrinsicObject">
				<rim:Name>
					<rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="RuleSet" />
				</rim:Name>
				<rim:Description>
					<rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="Represents a set of portrayal rules for use together."/>
				</rim:Description>
			</rim:ClassificationNode>

			<rim:ClassificationNode code="Rule"
			  objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ClassificationNode"
			  lid="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Rule"
			  id="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Rule"
			  parent="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ExtrinsicObject">
				<rim:Name>
					<rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="Rule" />
				</rim:Name>
				<rim:Description>
					<rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="Represents a portrayal rule."/>
				</rim:Description>
			</rim:ClassificationNode>

			<rim:ClassificationNode code="ApplicationSchema"
			  objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ClassificationNode"
			  lid="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:ApplicationSchema"
			  id="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:ApplicationSchema"
			  parent="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ExtrinsicObject">
				<rim:Name>
```

```xml
                                        <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="ApplicationSchema" />
                                </rim:Name>
                                <rim:Description>
                                        <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="Represents an application schema/feature catalogue."/>
                                </rim:Description>
                        </rim:ClassificationNode>

                        <rim:ClassificationNode code="SymbolSet"
                         objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ClassificationNode"
                         lid="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:SymbolSet"
                         id="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:SymbolSet"
                         parent="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ExtrinsicObject">
                                <rim:Name>
                                        <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="SymbolSet" />
                                </rim:Name>
                                <rim:Description>
                                        <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="Represents a set of related symbols."/>
                                </rim:Description>
                        </rim:ClassificationNode>

                        <rim:ClassificationNode code="Symbol"
                         objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ClassificationNode"
                         lid="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Symbol"
                         id="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Symbol"
                         parent="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ExtrinsicObject">
                                <rim:Name>
                                        <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="Symbol" />
                                </rim:Name>
                                <rim:Description>
                                        <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="Represents a symbol (point/icon, line-style, fill-style or text presentation) that can be used by
portrayal rules."/>
                                </rim:Description>
                        </rim:ClassificationNode>

                        <rim:ClassificationNode code="Font"
                         objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ClassificationNode"
                         lid="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Font"
                         id="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Font"
                         parent="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ExtrinsicObject">
                                <rim:Name>
                                        <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="Font" />
                                </rim:Name>
                                <rim:Description>
                                        <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="Represents a type font that can be used by portrayal rules and symbols."/>
                                </rim:Description>
                        </rim:ClassificationNode>

                        <rim:ClassificationNode code="Color"
```

```
                    objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ClassificationNode"
                    lid="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Color"
                    id="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Color"
                    parent="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ExtrinsicObject">
                        <rim:Name>
                            <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="Color" />
                        </rim:Name>
                        <rim:Description>
                            <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="Represents a named color that can be used by portrayal rules or symbols."/>
                        </rim:Description>
                    </rim:ClassificationNode>

                    <!-- Declaration of SymbolType taxonomy used to classify Symbol instances. -->
                    <rim:ClassificationScheme
                            lid="urn:x-ogc:specification:csw-ebrim:DPRS:SymbolTypes"
                            id="urn:x-ogc:specification:csw-ebrim:DPRS:SymbolTypes"
                            objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ClassificationScheme"
                            isInternal="true"
                            nodeType="urn:oasis:names:tc:ebxml-
regrep:NodeType:UniqueCode">
                        <rim:Name>
                            <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
value="SymbolTypes"/>
                        </rim:Name>

                        <rim:Description>
                            <rim:LocalizedString xml:lang="en-US" charset="UTF-8"
                                        value="This is the canonical
ClassificationScheme for the symbol types hierarchy."/>
                        </rim:Description>

                        <rim:ClassificationNode code="LineStyleSymbol"
                                objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ClassificationNode"
                                lid="urn:x-ogc:specification:csw-
ebrim:DPRS:SymbolTypes:LineStyleSymbol"
                                id="urn:x-ogc:specification:csw-
ebrim:DPRS:SymbolTypes:LineStyleSymbol"
                                parent="urn:x-ogc:specification:csw-
ebrim:DPRS:SymbolTypes">
                            <rim:Name>
                                <rim:LocalizedString xml:lang="en-US"
charset="UTF-8" value="LineStyleSymbol" />
                            </rim:Name>
                            <rim:Description>
                                <rim:LocalizedString xml:lang="en-US"
charset="UTF-8" value="Symbol describing a line presentation."/>
                            </rim:Description>
                        </rim:ClassificationNode>

                        <rim:ClassificationNode code="FillStyleSymbol"
                                objectType="urn:oasis:names:tc:ebxml-
regrep:ObjectType:RegistryObject:ClassificationNode"
                                lid="urn:x-ogc:specification:csw-
ebrim:DPRS:SymbolTypes:FillStyleSymbol"
                                id="urn:x-ogc:specification:csw-
ebrim:DPRS:SymbolTypes:FillStyleSymbol"
```

```
							parent="urn:x-ogc:specification:csw-ebrim:DPRS:SymbolTypes">
						<rim:Name>
							<rim:LocalizedString xml:lang="en-US" charset="UTF-8" value="FillStyleSymbol" />
						</rim:Name>
						<rim:Description>
							<rim:LocalizedString xml:lang="en-US" charset="UTF-8" value="Symbol describing a polygon fill style."/>
						</rim:Description>
					</rim:ClassificationNode>

					<rim:ClassificationNode code="PointSymbol"
							objectType="urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject:ClassificationNode"
							lid="urn:x-ogc:specification:csw-ebrim:DPRS:SymbolTypes:PointSymbol"
							id="urn:x-ogc:specification:csw-ebrim:DPRS:SymbolTypes:PointSymbol"
							parent="urn:x-ogc:specification:csw-ebrim:DPRS:SymbolTypes">
						<rim:Name>
							<rim:LocalizedString xml:lang="en-US" charset="UTF-8" value="PointSymbol" />
						</rim:Name>
						<rim:Description>
							<rim:LocalizedString xml:lang="en-US" charset="UTF-8" value="Symbol describing a point object or icon."/>
						</rim:Description>
					</rim:ClassificationNode>

					<rim:ClassificationNode code="TextSymbol"
						objectType="urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject:ClassificationNode"
						lid="urn:x-ogc:specification:csw-ebrim:DPRS:SymbolTypes:TextSymbol"
						id="urn:x-ogc:specification:csw-ebrim:DPRS:SymbolTypes:TextSymbol"
						parent="urn:x-ogc:specification:csw-ebrim:DPRS:SymbolTypes">
						<rim:Name>
							<rim:LocalizedString xml:lang="en-US" charset="UTF-8" value="TextSymbol" />
						</rim:Name>
						<rim:Description>
							<rim:LocalizedString xml:lang="en-US" charset="UTF-8" value="Symbol describing a textual presentation."/>
						</rim:Description>
					</rim:ClassificationNode>
				</rim:ClassificationScheme>

				<!-- Meta-RegistryObject declarations
				 (declare available Slots on instances of the new types of ExtrinsicObject added in this package) -->
				<rim:RegistryObject id="RuleSetId" objectType="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:RuleSet">
					<rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:previewImage" slotType="Image"><rim:ValueList/></rim:Slot>
					<rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:supportedEncodings" slotType="string"><rim:ValueList/></rim:Slot>
				</rim:RegistryObject>
```



```
<rim:RegistryObject id="RuleId" objectType="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Rule">
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:featureTypes" slotType="string"><rim:ValueList/></rim:Slot>
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:nativeEncoding" slotType="string"><rim:ValueList/></rim:Slot>
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:supportedEncodings" slotType="string"><rim:ValueList/></rim:Slot>
</rim:RegistryObject>

<rim:RegistryObject id="ApplicationSchemaId" objectType="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:ApplicationSchema">
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:namespace" slotType="string"><rim:ValueList/></rim:Slot>
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:featureTypes" slotType="string"><rim:ValueList/></rim:Slot>
</rim:RegistryObject>

<rim:RegistryObject id="SymbolSetId" objectType="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:SymbolSet">
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:symbols" slotType="referenceURI"><rim:ValueList/></rim:Slot>
</rim:RegistryObject>

<rim:RegistryObject id="SymbolId" objectType="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Symbol">
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:previewImage" slotType="Image"><rim:ValueList/></rim:Slot>
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:nativeEncoding" slotType="string"><rim:ValueList/></rim:Slot>
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:supportedEncodings" slotType="string"><rim:ValueList/></rim:Slot>
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:areaOfApplication" slotType="string"><rim:ValueList/></rim:Slot>
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:intendedUse" slotType="string"><rim:ValueList/></rim:Slot>
</rim:RegistryObject>

<rim:RegistryObject id="FontId" objectType="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Font">
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:previewImage" slotType="Image"><rim:ValueList/></rim:Slot>
</rim:RegistryObject>

<rim:RegistryObject id="ColorId" objectType="urn:x-ogc:specification:csw-ebrim:ObjectType:DPRS:Color">
    <rim:Slot name="urn:ogc:def:ebRIM-Slot:DPRS:previewImage" slotType="Image"><rim:ValueList/></rim:Slot>
</rim:RegistryObject>
</rim:RegistryObjectList>

</rim:RegistryPackage>
```


7.3 FPS

A Feature Portrayal Service (FPS) was supplied by Carmenta.

7.4 WFS

Web Feature Service (WFS) instances were supplied by:

1. CubeWerx
2. Interactive Instruments

8 Interoperability Issues and Resolutions

This section summarizes some of the issues and recommendations arising from the TIE.

1. The ability to carry data (as in KML <ExtendedData> [6]) should be added to SE Symbolizers so that feature property values can be used in symbols.

2. Feature type names referenced in FeatureType Styles (SE or otherwise) must be provided as fully qualified names (i.e. with namespaces).

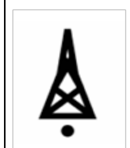
3. An ebRIM model for Portrayal can be created which is abstract enough to enable multiple rule & symbol encodings, and which provides sufficient visible (in the registry)information to support 1) creation of styling documents (SLD) and 2) discovery of symbols, and styles appropriate to the presentation of particular feature types, and 3) management of symbol libraries.

[6] Some believe that this is sufficient to make KML feature-oriented.

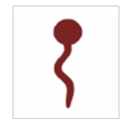
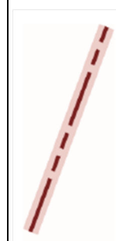
Annex A

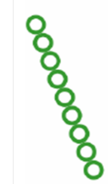
OGC Symbology Encoding Support for LTDS

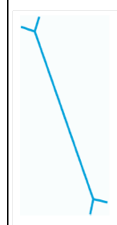
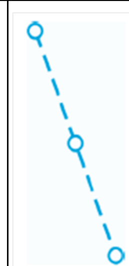
A.1 Analysis of SE Support for LTDS

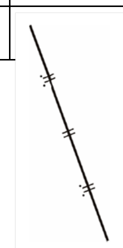
Feature Type	Geom[7]	LTDS Element Name	Element Reference	LTDS Symbol Description	SLD	FPS	Technical Notes	Graphic	SLD/SE Fragment
Aerial (AT011)	P	Black_Communication_Tower	P00155	Line weight (all): 0.20mm Dot 0.4mm circle with a gap of 0.2mm between edge of dot and edge of tower base. Lattice: Bottom corner to 1.0 mm up along edge 1.0mm up edge to 2.0 mm up along edge. Angles: Base to "Mid": 70 degrees, "Mid" to "Top": 81.5 degrees, Origin: Centre of dot	Yes	Yes	<PointSymbolizer/> used to draw an icon at a point. <Graphic/> element is used to point to either vector or raster source type. The center used for positioning the rendering at a point must either be inherent in the external format, or the <AnchorPoint/> element of <PointSymbolizer/> can be used to anchor the graphic to the main-geometry point.		<PointSymbolizer> <Geometry> <ogc:PropertyName>aml:position</ogc:PropertyName> </Geometry> <Graphic> <ExternalGraphic> <OnlineResource xlink:href="localhost/Symbols/LTDSP0155.png" xlink:title="LTDSP00155"/> <Format>image/png</Format> </ExternalGraphic> <Size> <ogc:Literal>8.0</ogc:Literal> </Size> <Rotation>0</Rotation>

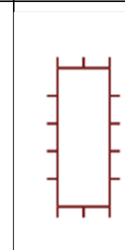
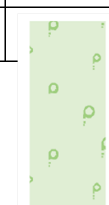
[7] Dimensionality of the symbol – P = Point, L = Line and A = Area

Feature Type	Geom[7]	LTDS Element Name	Element Reference	LTDS Symbol Description	SLD	FPS	Technical Notes	Graphic	SLD/SE Fragment
									</Graphic> </PointSymbolizer>
GeothermalOutlet (DB115)	P	Dk-Brown1815_Geothermal	P00027	Tail of feature is downstream to drainage or to the lines of latitude when drain is not present	No	No	SLD/SE does not currently support directional indicators. Therefore, if the tail is directional, this symbol is not supported by the current SLD/SE specification.		Not Applicable
AdministrativeBoundary (FA000)	L	Dk-Brown1815_AdminBoundary_Other	L00254	Dk-Brown1815 0.5mm Dashed line (0.5mm wt; Dash1: 2.0mm Gap1: 1.0mm; Dash2: 2.0mm Gap2: 1.0mm; Dash3: 7.0mm Gap3: 1.0mm) with a Dk-Brown1815-42 overprint	Yes	Yes	Currently displayed using <LineSymbolizer/> css parameters only. In particular, the stroke-dasharray CssParameter element encodes a dash pattern as a series of space separated floats. The first number gives the length in pixels of dash to draw, the second gives the amount of space to leave, and this pattern repeats. The default is to draw an unbroken line.		<!-- for Dk-Brown1815-42_2.0mmScreenedLine Element Reference: L00126 --> <LineSymbolizer> <Geometry> <ogc:PropertyName>dafif:centreLine</ogc:PropertyName> </Geometry> <Stroke> <CssParameter name="stroke"> <ogc:Literal>rgb(234,200,195)</ogc:Literal> </CssParameter> <CssParameter name="stroke-width"> <ogc:Literal>2.0</ogc:Literal> </CssParameter> </Stroke>

Feature Type	Geom[7]	LTDS Element Name	Element Reference	LTDS Symbol Description	SLD	FPS	Technical Notes	Graphic	SLD/SE Fragment
									</LineSymbolizer> <!-- for Dk-Brown1815_0.5mm-2-1-2-1-7-1DashLine Element Reference: L00253 --> <LineSymbolizer> <Geometry> <ogc:PropertyName>dafif:centreLine</ogc:PropertyName> </Geometry> <Stroke> <CssParameter name="stroke"> <ogc:Literal>rgb(120,35,39)</ogc:Literal> </CssParameter> <CssParameter name="stroke-width"> <ogc:Literal>0.5</ogc:Literal> </CssParameter> <CssParameter name="stroke-dasharray"> <ogc:Literal>2.0 1.0 2.0 1.0 7.0 1.0</ogc:Literal> </CssParameter>
Forest (EC015)	L	Green362_Hedgerow	L00060	Green362_0.1mm OpenCircle circles repeating along the linear feature.	Yes	Yes	SLD/SE supports this symbol via <GraphicStroke/> element of <LineSybolizer/>.		<LineSymbolizer> <Geometry> <ogc:PropertyName>dafif:centreLine</ogc:PropertyName> </Geometry> <Stroke> <GraphicStroke> <Graphic> <ExternalGraphic> <OnlineResource xlink:href="localhost/Symbols/LN_Green362_Forest.png"

Feature Type	Geom[7]	LTDS Element Name	Element Reference	LTDS Symbol Description	SLD	FPS	Technical Notes	Graphic	SLD/SE Fragment
									xlink:title="LNGREEN362FORESTt"/> <Format>image/png</Format> </ExternalGraphic> <Size> <ogc:Literal>0.5</ogc:Literal> </Size> <Rotation>0</Rotation> </Graphic> </GraphicStroke> </Stroke> </LineSymbolizer>
Aqueduct (BH010)	L	Cyan_Aqueduct / Penstock	L00106	Cyan 0.3mm Solid Line with 'V' (0.3mm wt, 0.7mm long)	No	No	If an SVG symbols is used, SVG marker properties 'marker-start' and 'marker-end' may enable support of this style, using the first and last linestring coordinate, coord, or gml:pos as the marker locations. marker-start and marker-end are supported on 'path', 'line', 'polyline' and 'polygon' elements. While the current SLD spec may support this complex style using multiple styling rules, specific geometry modeling may be needed to support it.		
Aqueduct (BH010)	L	Cyan_Aqueduct_with_Qanats	L00249	Cyan 0.2mm Dash Line (0.2mm wt, 1.0mm dash, 0.5mm gap) with repeating Cyan open circles (0.15mm wt, 1.0mm diameter, paperwhite fill)	Yes	Yes			<!-- Cyan_0.2mm-1mmLenDash-0.5mmGapLine L00040 --> <LineSymbolizer> <Geometry> <ogc:PropertyName>dafif:centreLine</ogc:PropertyName> </Geometry> <Stroke> <CssParameter name="stroke"><ogc:Literal>rgb(0,159,218)</ogc:Literal> </CssParameter>

Feature Type	Geom[7]	LTDS Element Name	Element Reference	LTDS Symbol Description	SLD	FPS	Technical Notes	Graphic	SLD/SE Fragment
									<CssParameter name="stroke-width"><ogc:Literal>0.2</ogc:Literal> </CssParameter> <CssParameter name="stroke-dasharray"> <ogc:Literal>1.0 0.5</ogc:Literal> </CssParameter> </Stroke> </LineSymbolizer> <LineSymbolizer> <Geometry> <ogc:PropertyName>dafif:centreLine</ogc:PropertyName> </Geometry> <Stroke> <GraphicStroke> <Graphic> <!-- Cyan_0.15mmCircleOpen P00057--> <ExternalGraphic> <OnlineResource xlink:href="localhost/Symbols/P00057.png" xlink:title="Cyan_0.15mmCircleOpen"/> <Format>image/png</Format> </ExternalGraphic> <Size> <ogc:Literal>0.15</ogc:Literal> </Size> </Graphic> </GraphicStroke> </Stroke> </LineSymbolizer>
Railway (AN010)	L	Black_Railway (Double Normal Gauge, Electric)	L00074	Black 0.3mm Solid Line with Double Ticks (0.15mm wt,	No	No	The double ticks could be achieved by specifying the		Not Applicable

Feature Type	Geom[7]	LTDS Element Name	Element Reference	LTDS Symbol Description	SLD	FPS	Technical Notes	Graphic	SLD/SE Fragment
				1.5mm long, 6.4mm interval) and 0.5mm Dots			rotation of the <Graphic/> element used in <GraphicStroke/> of the <LineSymbolizer/>. However the initial absolute angle needs to be specified so that the double ticks are perpendicular to the main line. The SLD/SE 1.1 does not specify this. It could be supported in SLD/SE 1.1 if the text of the standard is clarified. The SVG 'marker-mid' property may allow this complex linestyle to be adequatly supported, however it styles all intermediate vertices. This means that sophisticated conversion of original geometries would likely be required to place the ticks at appropriate intervals and perpendicular to the centre line.		
Embankment (DB090)	L	Dk-Brown1815_Embankment-Outline	L00101	Dk-Brown1815 0.15mm Solid Line with Ticks (0.15mm wt, 0.5mm long)	No	No			Not Applicable
Forest (EC015)	A	Woodland-Deciduous	A00038	Green362-31_ScreenFill (area fill) coincident with	Yes	Yes	Green362_DeciduousTreeFill filled using repeated		<!-- Green362-31_ScreenFill --> <PolygonSymbolizer>

Feature Type	Geom[7]	LTDS Element Name	Element Reference	LTDS Symbol Description	SLD	FPS	Technical Notes	Graphic	SLD/SE Fragment
				Green362_DeciduousTreeFill (area pattern)			<GraphicFill/> with <Opacity/> set to 0.5 (50%). Green362-31_ScreenFill filled using solid fill with SvgParameter fill-opacity set to 0.5 (50%).		<Geometry> <ogc:PropertyName>the_area</ogc:PropertyName> </Geometry> <Fill> <SvgParameter name="fill">#a8e4c6</SvgParameter> <SvgParameter name="fill-opacity">0.5</SvgParameter> </Fill> </PolygonSymbolizer> <!--Green362_DeciduousTreeFill --> <PolygonSymbolizer> <Geometry> <ogc:PropertyName>the_area</ogc:PropertyName> </Geometry> <Fill> <GraphicFill> <Graphic> <ExternalGraphic> <OnlineResource xlink:href="localhost/Symbols/A00011.png" xlink:title="A00011"/> <Format>image/png</Format> </ExternalGraphic> <Opacity>0.5</Opacity> </Graphic> </GraphicFill> </Fill> </PolygonSymbolizer>

Bibliography

[1] OGC® OWS-5 GSIP Schema Processing Engineering Report, OGC document 08-078r1

[2] OGC® OWS-6 GML Profile Validation Tool Engineering Report, OGC document 09-038r1

[3] OGC® OWS-7 Engineering Report - Aviation Portrayal, OGC document 10-127r1

[4] OGC® OWS-7 Schema Automation Engineering Report, OGC document 10-088r2

[5] OGC® The Specification Model — Modular specifications, OGC document 08-131r3

[6] OGC® Web Feature Service, version 2.0.0, OGC document 09-025r1 (OGC standard)

[7] OGC® Styled Layer Descriptor profile of the Web Map Service Implementation Specification, OGC document 05-078r4, dated 2007-06-29.

[8] ISO TC/211 Rules for Application Schemas DIS ISO 19109. Dated 2002.

[9] OGC® Symbology Encoding Implementation Specification. OGC Document 05-077r4, dated 2006-07-21

[10] Proposed Portrayal Registry Extension Package Specification, Version PA5, 2011-05-11