Open GIS Consortium Inc.

Date: 2002-01-20

Reference number of this OpenGIS© IP initiative document: **OGC 03-031**

Version: 0.0.9

Category: OpenGIS© Interoperability Program Report

Editor: William Lalonde

Style Management Service (SMS) IPR

Copyright notice

Warning

This document is not an OGC Standard or Specification. This document presents a discussion of technology issues considered in an Interoperability Initiative of the OGC Interoperability Program. The content of this document is presented to create discussion in the geospatial information industry on this topic; the content of this document is not to be considered an adopted specification of any kind. This document does not represent the official position of the OGC nor of the OGC Technical Committee. It is subject to change without notice and may not be referred to as an OGC Standard or Specification. However, the discussions in this document could very well lead to the definition of an OGC Implementation Specification.

Recipients of this document 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: OpenGIS© Interoperability Program Report
Document stage: Draft
Document language: English

FileName: 03-031.doc

Contents

i. Preface

This Interoperability Program Report (IPR) document was developed as part of the Feature Handling Thread of the OGC Web Services testbed Phase 1 Thread Set 2 (OWS-1.2). This version of the IPR reflects the status of the specification after the main public demonstration at Lockheed-Martin's Gaithersburg facilities in November 2002 and replaces all previous Draft (DIPR) versions of the document.

ii. Submitting organizations

The following organizations submitted this document to the Open GIS Consortium Inc.

CubeWerx Inc.

iii. Document Contributor Contact Points

All questions regarding this submission should be directed to the editor or the submitters:

COMPANY	CONTACT	ADDRESS	PHONE/FAX	EMAIL
CubeWerx Inc.	William Lalonde	200 rue Montcalm Suite R-13 Hull, Quebec Canada J8Y 3B5	Phone: 819-771-8303 Ext 206 Fax: 819-771-8388	wlalonde@cubewerx.com
CubeWerx Inc.	Craig Bruce	200 rue Montcalm Suite R-13 Hull, Quebec Canada	Phone: 819-771-8303 Ext 205 Fax:	csbruce@cubewerx.com

		J8Y 3B5	819-771-8388	
Galdos Systems Inc.	Milan Trninic			mtrninic@galdosinc.com
Ionic Software s.a.	Dimitri Monie			dimitri.monie@ionicsoft.com
Syncline	Josh Lieberman			josh@syncline.com
Polexis	Jeff Lansing			jeff@polexis.com
Polexis	Chris Dillard			cdillard@polexis.com
U of A - CAST	Shane Covington			scoving@cast.uark.edu

iv. Revision history

Date	Release	Author	Paragraph modified	Description
2002-07-04	0.0.1	Craig Bruce William Lalonde	all	Initial Release
2002-07-19	0.0.2	Craig Bruce William Lalonde	all	Cosmetic updates to initial release.
2002-07-23	0.0.3	Craig Bruce William Lalonde Jeff Lansing	7.1, 7.3 and Annex A	Add pointer to V0.7.3 SLD schemas plus add actor diagrams for “Using SMS”.
2002-07-31	0.0.4	Craig Bruce William Lalonde Chris Dillard	7.2	Add sequence diagram.
2002-08-19	0.0.5	Craig Bruce William Lalonde Chris Dillard	Annex C	Add style and symbol metadata schemas.
2003-01-06	0.0.6	Craig Bruce William Lalonde	Various	Added additional informative text: Preface, Forward, Introduction, etc. Added an example of SLD using the SMS.

2003-01-14	0.0.7	Craig Bruce William Lalonde Shane Covington	Annex C	Updated discussion of metadata
2003-01-15	0.0.8	Craig Bruce William Lalonde Shane Covington	4, 6.2.4, Annex C	Updated discussion of metadata
2003-01-16	0.0.9	William Lalonde	4, 6.2.5.1, 6.2.5.2, 6.2.5.3, 6.2.5.4. 7.2	Minor editorial updates.

v. Changes to the OpenGIS® Abstract Specification

The OpenGIS© Abstract Specification does not require changes to accommodate the technical contents of this document.

vi. Future Work

The contents of this specification should maintain alignment with OGC Document 02-070 Styled Layer Descriptor (SLD V1.0.0) Specification (or its successor document) and may result in future changes to the SLD Specification.

Foreword

Attention is drawn to the possibility that some of the elements of this part of OGC 02-046r9 may be the subject of patent rights. The Open GIS Consortium Inc. shall not be held responsible for identifying any or all such patent rights.

This is the sixth version of this specification and the first to be released without a Draft (DIPR) status. This document replaces the fifth and all previous versions of this specification.

Introduction

This document describes the proposed system design for the OGC Style Management Service (SMS).

The SMS must manage distinct objects that represent styles and symbols and provide the means to discover, query, insert, update, and delete these objects.

Styles provide the mapping from feature types and feature properties and constraints to parameterized Symbols used in drawing maps. Symbols are bundles of predefined graphical parameters and predefined fixed graphic "images".

OGC Style Management Service (SMS)

1 Scope

This document addresses the requirements for the Style Management Service as stated in the most recent Statement of Requirements [5]. This specification reflects the implementation of the SMS as it existed at the time of the OWS-1.2 public demonstration in November 2002.

2 Conformance

Not required for an IP IPR, DIPR, or Discussion Paper.

3 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of OGC 02-046. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of OGC 02-046 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies.

CGI, *The Common Gateway Interface,* National Center for Supercomputing Applications, http://hoohoo.ncsa.uiuc.edu/cgi/

IETF RFC 2045 (November 1996), *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodie*s, Freed, N. and Borenstein N., eds., http://www.ietf.org/rfc/rfc2045.txt

IETF RFC 2616 (June 1999), *Hypertext Transfer Protocol – HTTP/1.*1, Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T., eds., http://www.ietf.org/rfc/rfc2616.txt

IETF RFC 2396 (August 1998), *Uniform Resource Identifiers (URI): Generic Syntax*,

Berners-Lee, T., Fielding, N., and Masinter, L., eds.,

OGC 02-112r2 , *The OpenGIS Abstract Specification Topic 12: OpenGIS Service Architecture (Version 4.3*), Percivall, G. (ed.), January 2002 http://www.opengis.org/techno/abstract/02-112.pdf

OGC 01-068r2 Adopted Implementation Specification: Web Map Server version 1.1.1, February 2002.

OGC 01-047r2 Adopted Implementation Specification: Web Map Server version 1.1.0, December 2001.

OGC 02-024 RFC Web Coverage Service version 0.7, November 2002.

XML 1.0 (October 2000), *Extensible Markup Language (XML) 1.0 (2nd edition*), World Wide Web Consortium Recommendation, Bray, T., Paoli, J., Sperberg-McQueen, C.M., and Maler, E., eds., <http://www.w3.org/TR/2000/REC-xml>
http://www.ietf.org/rfc/rfc2396.txt

4 Terms and definitions

4.1
operation
Specification of a transformation or query that an object may be called to execute [13]

4.2
interface
Named set of **operations** that characterize the behavior of an entity [13]

4.3
service
Distinct part of the functionality that is provided by an entity through **interfaces** [13]

4.4
service instance
server
Actual implementation of a **service**

4.5
client
Software component that can invoke an operation from a server

4.6
style
Styles provide the mapping from feature types and feature properties and constraints to parameterized Symbols used in drawing maps.

4.7
symbol
Symbols are bundles of predefined graphical parameters and predefined fixed graphic "images".

5 Conventions

5.1 Symbols (and abbreviated terms)

CGI - Common Gateway Interface
DCP - Distributed Computing Platform
DTD - Document Type Definition
EPSG - European Petroleum Survey Group
GIF - Graphics Interchange Format
GIS - Geographic Information System
GML - Geography Markup Language
HTTP - Hypertext Transfer Protocol
IETF - Internet Engineering Task Force
JPEG - Joint Photographic Experts Group
MIME - Multipurpose Internet Mail Extensions
OGC - Open GIS Consortium
OWS - OGC Web Service
PNG - Portable Network Graphics
RFC - Request for Comments
SLD - Styled Layer Descriptor
SVG - Scalable Vector Graphics
URL - Uniform Resource Locator
WebCGM - Web Computer Graphics Metafile
WCS - Web Coverage Service
WFS - Web Feature Service
WMS - Web Map Service
XML - Extensible Markup Language

5.2 UML Notation

The diagrams that appear in this document are presented using the Unified Modeling Language (UML) static structure diagram. The UML notations used in this document are described in the diagram below.

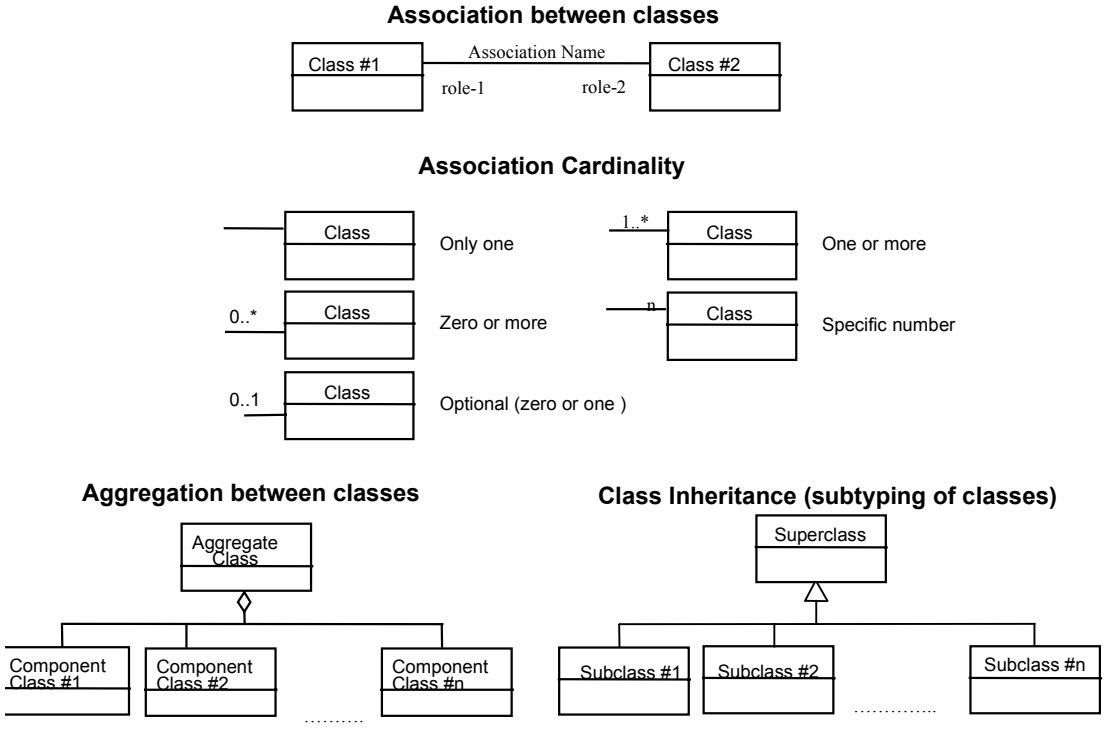


Figure 1 — UML notation

In this diagram, the following three stereotypes of UML classes are used:

a) <<Interface>> A definition of a set of operations that is supported by objects having this interface. An Interface class cannot contain any attributes.

b) <<DataType>> A descriptor of a set of values that lack identity (independent existence and the possibility of side effects). A DataType is a class with no operations whose primary purpose is to hold the information.

c) <<CodeList>> is a flexible enumeration that uses string values for expressing a list of potential values.

6 Style Management Service Design

6.1 System architecture

The SMS system has two major components that can function independently: a Style Manager and a Symbol Manager. Each of these components comprise two sub-components: an OGC Registry Service [11] and an OGC Repository [12]. Both of these sub-components are generic (may be used for any purpose) and are defined by evolving OGC specifications. Figure 1 provides a UML diagram of the architecture.

The purpose of a registry is to manage metadata about extrinsic data (or service) objects and to provide an interface for searching and discovery.

An OGC Repository is still an evolving component. It is currently based on the OGC Web Object Service [12] and it stores and provides an interface to access data objects.

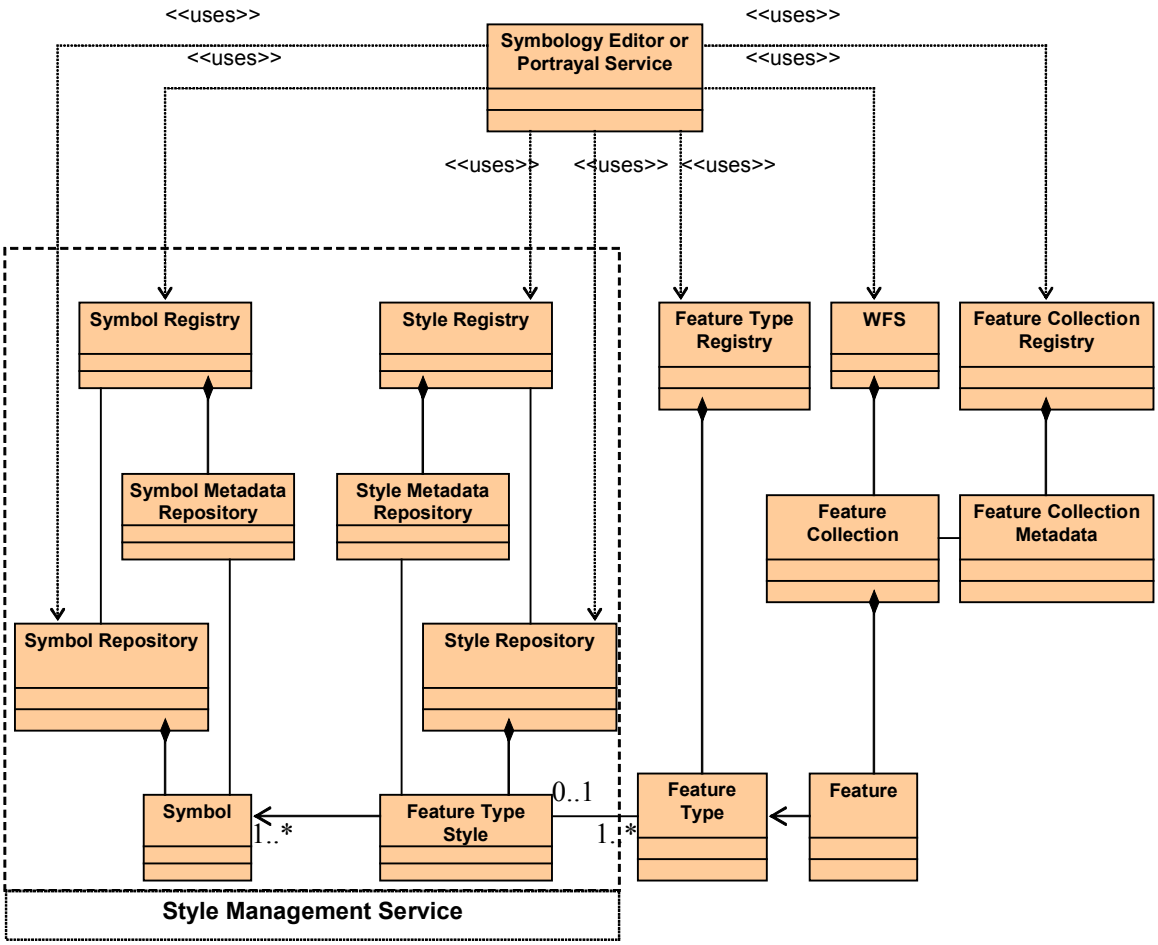


Figure 1 - UML Notational Form of SMS System Architecture

6.2 System design

6.2.1 SLD, style and symbol representation

In principle, any number of representations may be used with the proposed SMS architecture. However, for the OWS-1.2 project, it is proposed that the primary representation be based on the OGC Styled Layer Descriptor Implementation Specification (SLD) [6]. The version currently in use is 1.0.0, which is an adopted specification. This document introduces changes to SLD 1.0.0 so that the SMS requirements set forth can be met. The changes made here are to be submitted as a change proposal to the SLD Revision Working Group.

A number of tag names have been changed for convenience and consistency with SMS and industry terminology. The `FeatureTypeStyle` tag has been renamed to just `FeatureStyle` for the sake of convenience, especially when speaking.

The term "Symbolizer" has been renamed to just "Symbol" to be consistent with the SMS architecture and what appears to be the common usage of the term 'symbol' in industry. The term "Symbolizer" was invented in a previous SLD version to reduce confusion between Symbols and SLD "Graphic" elements, but the neologism may have caused more confusion. All of the tags that contained the term "Symbolizer" have been correspondingly renamed.

Finally, the `CssParameter` tag has been rename to `SvgParameter`. Referring to style parameters as CSS-related is technically incorrect. Parameters such as "`stroke-width`" are part of the SVG specification and not the CSS specification. The SVG language includes CSS.

SLD 1.0.0structure has been modified for use in the SMS environment. SLD was primarily designed for use in the Web Map Service (WMS) [9] environment. An abstract overview of the structure of SLD 1.0.0 (with the above renamings) is:

```
<StyledLayerDescriptor>
   <NamedLayer> or <UserLayer>
      <UserStyle>
         <FeatureStyle>
            <Rule>
               <LineSymbol> or ... or <RasterSymbol>
                   ... parameter content ...
                      <Graphic>
```

This monolithic structure has been split into four different components (schema files):

"StyledLayerDescriptor.xsd", "FeatureStyle.xsd", "Symbol.xsd", and "common.xsd". The first three schema files represent the hierarchy of SLD objects, and the last file is a catch-all for tags that are used throughout the other files.

The "StyledLayerDescriptor.xsd" schema contains the SLD elements above the split between the `UserStyle` and `FeatureStyle` XML-element levels. The elements above this split are not truly associated with styling but are instead geared towards defining WMS "layers" which is a concept specific to WMS. These elements therefore should be used only in association with the WMS and WMS-derived interfaces. In fact, they already are, in the HTTP-POST WMS version designed in the OWS-1.1 project [10]. The `GetLegendGraphic`, `GetStyle` and `PutStyle` operations defined for SLD should also be a part of only the WMS interface, since they deal with full SLDs and layers and storing & retrieving & using 'named' styling information from WMSes.

The “FeatureStyle.xsd” schema includes the elements at the XML-element levels between `FeatureStyle` and `Rule`. This contains the actual styling directives, so styling has been re-focused exclusively on feature types (and equivalents), with no concept of ‘layers’. This schema can be used independently from the WMS-oriented “StyledLayerDescriptor.xsd” schema in order to handle styles independently from SLDs.

The “Symbol.xsd” schema includes all of the symbols. This schema can be used indendently from the “FeatureStyle.xsd” layer in order to handle symbols as independent and complete objects and to create a reusable symbol library.

A `UseSLDLibrary` sub-element has been added to `StyledLayerDescriptor` so that external SLD documents can be used in “library-mode” even when using XML-encoded requests with a WMS. (The “library mode” can be accessed with the HTTP-GET method by supplying an `SLD` CGI parameter in addition to `LAYERS` and `STYLES` CGI parameters.) This addition merely exercises pre-existing functionality in WMS, but it does add the wrinkle of making SLD-library references iterative and (syntactically) recursive.

An `EnvironmentVariables` ‘container’ element also has been added to `StyledLayerDescriptor` so that environment variables may be declared using a ‘blob’ of XML. Either the Filter spec needs to be modified to reference environment variables distinctly from feature properties, or XML namespaces can be used for this purpose, when the environment-variable XML blob is given a distinct namespace. These environment variables can be used to pass parameters to a style. These parameters are in addition to the properties of the features that a style will render.

The direct and repeated usage of `Title` and `Abstract` tags has been replaced with using a `Description` element which now includes the `Title` and `Abstract` sub-tags. This organization is more abstracted, reusable, and extensible for the future. The `Name` element is not included in the `Description` because it has a slightly different semantic from merely being descriptive.

A version attribute has been added to the `FeatureStyle` and the `Symbol` tags so that the version numbers can be identified when the SLD pieces are used independently.

An `OnlineResource` alternative has been added to the `FeatureStyle` tag to allow referencing external feature styles, such as those stored in a style repository.

An issue with the symbol repository is that the symbols in it must be stored without any references to feature properties, since they must be completely decoupled from specific feature types in order to be reusable. This means that the symbol-reference mechanism embedded within the `FeatureStyle/Rule` must be able to supply parameters to the symbol that are able to reference the feature properties.

This is accomplished using a symbol-parameter-value-override mechanism. A `BaseSymbol` element has been added to the (abstract) `Symbol` element to allow referencing & extension of external symbols, which may be stored in a symbol repository. The `BaseSymbol` element contains an `OnlineResource` sub-element for the external reference. Here is a highway-symbol example:

```
<FeatureStyle>
  ...
  <Rule>
    ...
    <LineSymbol>
      <BaseSymbol>
        <OnlineResource xlink:type="simple"
xlink:href="http://some.site.com/sym_repository.php?request=GETOB
JECT&objectid=some_sym_123"/>
      </BaseSymbol>
      <Geometry>
        <ogc:PropertyName>THE_ROAD_CENTERLINE</ogc:PropertyName>
      </Geometry>
      <Stroke>
        <SvgParameter name="stroke-width">
          <ogc:Div>
            <ogc:PropertyName>NUMBER_OF_LANES</ogc:PropertyName>
            <ogc:Literal>2</ogc:Literal>
          </ogc:Div>
        </SvgParameter>
      </Stroke>
    </LineSymbol>
  </Rule>
</FeatureStyle>
```

The semantics of using a `BaseSymbol` are that the referenced external symbol is read in and then the in-line graphical properties override the ones read in from the base symbol. In the above example, the external base symbol will be read in and then its "`stroke-width`" graphical parameter will be changed to the number of lanes divided by two. The base symbol should not include any references to any feature properties that might tie the symbol to any specific feature type, in order to maximize the potential reusability of library symbols.

An `InlineContent` element was added in order to allow `Graphic` content to be put in-line (as opposed to the `OnlineResource` element). It has an `encoding` attribute to specify if the encoding is in-line "`xml`" or "`base64`"-encoded binary. The data-content format is given as part of the `ExternalGraphic` tag. One minor issue is that the parent-tag name "`ExternalGraphic`" may not be the most suitable any more.

The enumerated type for `Raster/OverlapBehavior` has been changed to be a restricted string-content type rather than empty elements like before. This seems to be more the norm for enumerated values in XML documents.

Finally, a `MappedColorSymbol` element has been added to allow a simple and efficient encoding of "choropleth" symbolization (color-by-property-value). The tag is called "`MappedColorSymbol`" instead of "`ChoroplethSymbol`" out of fear that few people would know what "choropleth" means or how to spell the word. The symbol works by redefining the default coloring of an embedded secondary symbol.

6.2.2 Style metadata

Style metadata is an object which describes extrinsic style objects. The metadata is represented as an XML document with an XML Schema. The metadata should include descriptive information such as titles and keywords and contact information such as defined in ISO 1911[579]. Annex C includes the schema for style metadata and a description of the elements it defines.

Each style will need name that can be used as a primary key to identify and access the style and metadata. The metadata will also need to contain foreign keys for all of the feature types to which a style can be applied. This allows an client program, after locating feature styles, to more easily location styles which may be used with it, or vice versa. Style references could also be stored in the Feature Type registry.

The capability to group styles into libraries is also needed. Possibly the Registry Service taxonomy mechanism can be used for this purpose.

6.2.3 Symbol metadata

Symbol metadata is an object which describes symbol objects. The metadata needs of symbols are analogous to those of styles. The metadata for symbols uses the same schema as for styles, and is described in Annex C.

6.2.4 Repositories

For symbol/style repositories, it was originally proposed to reuse the WFS interface. To avoid confusion with the term "feature", a generalization has been made to call the derived service Web Object Service (WOS). The WOS essentially renames the term "feature" to "object" in every place in the interface where it is referenced. The purpose of the repositories is to store the relevant objects (XML blobs) and to allow access for Query, Insert, Update, and Delete to those objects. A simple URL reference can be used for access when the object id is known, as with the WFS interface:

http://some.repository.com/wos.php?request=GETOBJECT&objectid=my_object_123

6.2.5 System components

Each of the four SMS system components, Style Registry, Style Repository, Symbol Registry, and Symbol Repository presents an externally accessible interface. The components may be grouped together within one implementation or they conceptually may be implemented individually. Maintaining the consistency of information between all four components may be a challenge if they are not implemented as a single integrated system.

6.2.5.1 Style registry

The purpose of the Style Registry is to manage metadata about style objects and enable discovery of these resources by clients (applications and/or other services)

6.2.5.2 Style repository

The purpose of the Style Repository is to store instances of style objects which can be accessed by clients (applications and/or services).

6.2.5.3 Symbol registry

The purpose of the Symbol Registry is to manage metadata about symbol objects and enable discovery of these resources by clients (applications and/or other services).

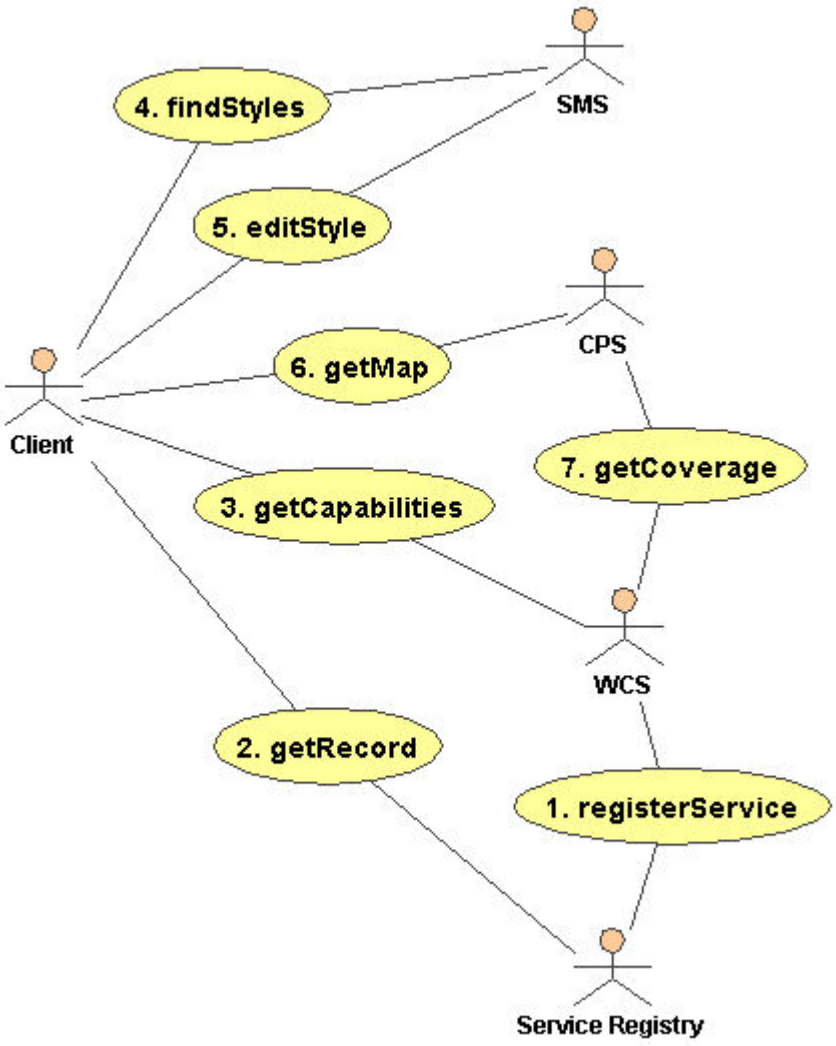
6.2.5.4 Symbol repository

The purpose of the Symbol Repository is to store instances of symbol objects which can be accessed by clients (applications and/or services).

7 Using the SMS

7.1 Basic scenario

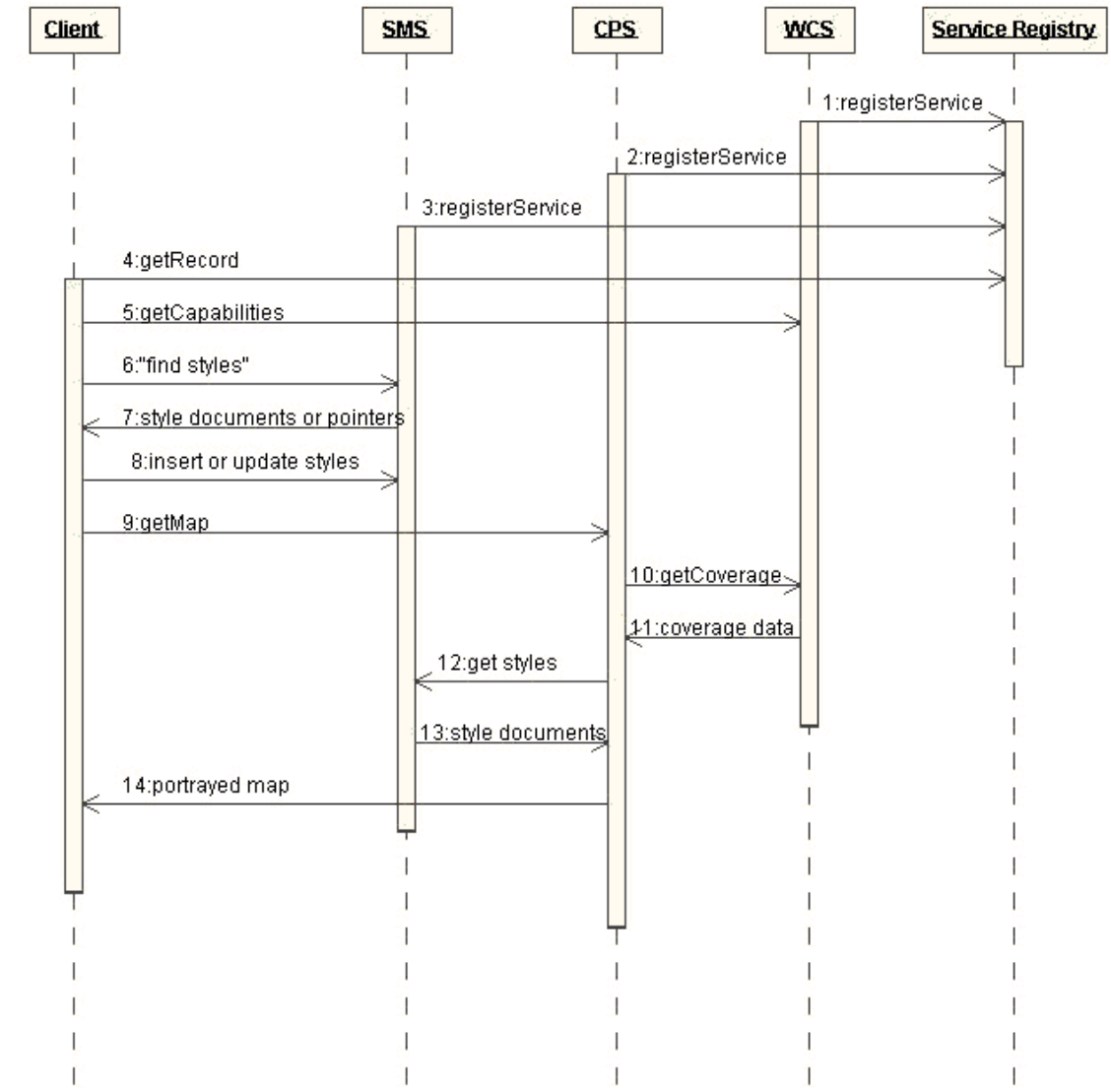
The following actor diagram shows one possible way to portray coverages using the SMS, without imposing too much of a burden on the client.



1. The scenario begins with the registering of a WCS with a Service Registry. Presumably this will allow the client to find services that provide desired types of data.
2. The client finds a service that supplies data of the desired type.
3. The client gets information about the data, in this case directly from the service.
4. The client chooses some data and asks the SMS what styles can handle that sort of data.
5. The client fills in some parameters in a style template, combines this with data from steps 2 and 3, and builds a SLD.
6. The client POSTs the SLD to a CPS, together with other parameters.
7. The CPS reads the SLD to find out about the WCS, combines parameters from the SLD and the GetMap request, and gets a coverage from the WCS.

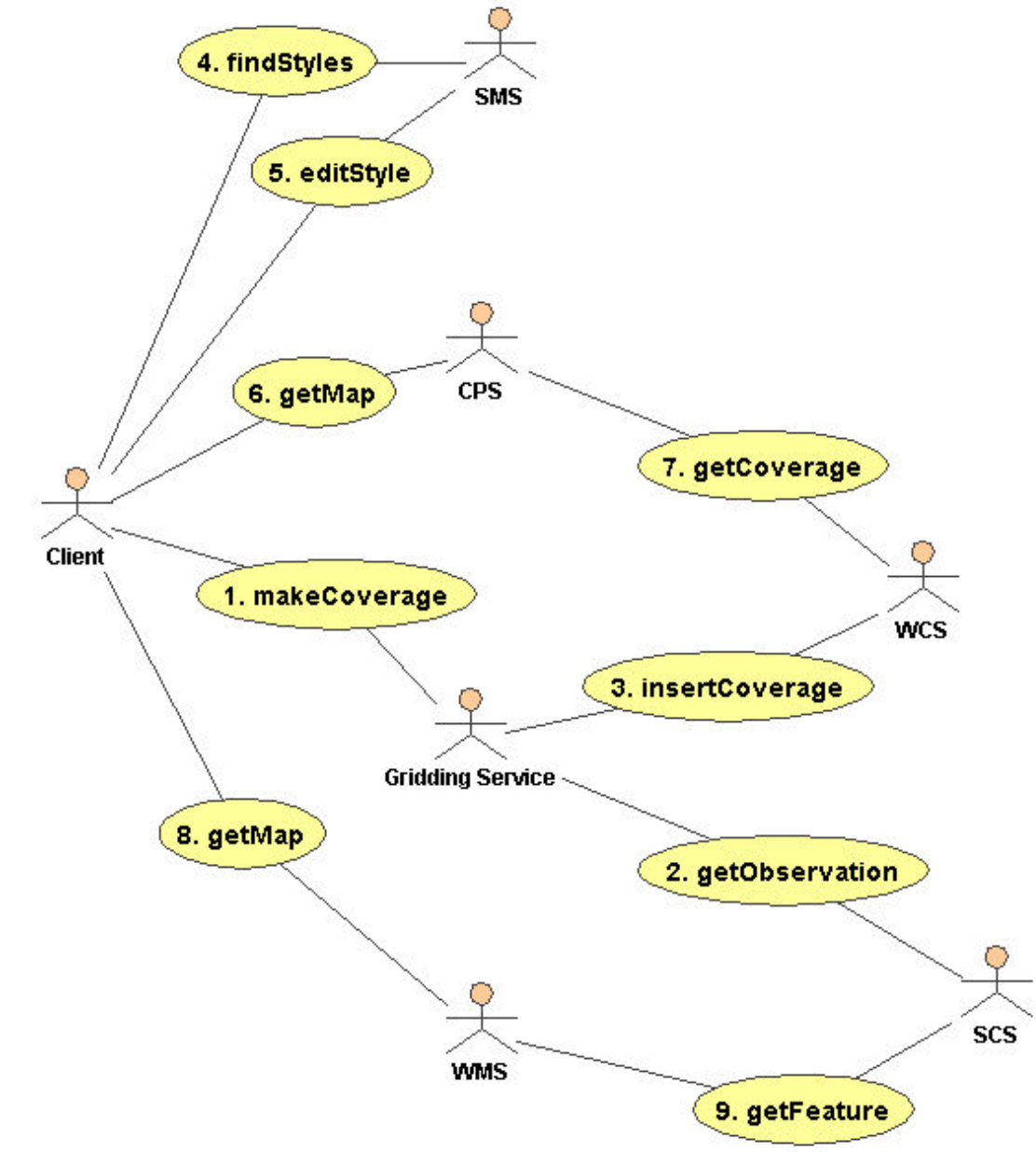
7.2 Sequence diagram

The following sequence diagram indicates the flow of data when using an SMS.



7.3 Extended scenario for non-gridded data from SCS

The extended scenario is this:



1. The client begins by asking a gridding service to make a coverage out of observation data. By some as yet unknown mechanism, perhaps involving something like SLD, the client induces the gridding service to
2. get the desired observations from a SCS, produce a grid using gridding parameters, and
3. insert the resulting grid, in some well-know coverage format, into a WCS.
4. As in the previous scenario, the client finds styles which are applicable, and
5. edits the parameters appropriately.
6. The client the sends a GetMap request to the CPS, which
7. gets the previously inserted coverage from the WCS, and portrays it. The result might look something like this:

Annex A
(normative)

Styled Layer Descriptor Schemas for SMS

A.1 General

Schemas for SMS compatible SLD (temporarily numbered V0.0.20) are available in an archive found on the OWS-1.2 portal. They are included in clause A.2.

A.2 Schemas

Common.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.opengis.net/sld"
            xmlns:sld="http://www.opengis.net/sld"
            xmlns:xlink="http://www.w3.org/1999/xlink"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            elementFormDefault="qualified">
  <xsd:import namespace="http://www.w3.org/1999/xlink"
              schemaLocation="../../gml/2.1/xlinks.xsd"/>

<!-- ********************************************************************** -->
  <xsd:annotation>
    <xsd:documentation>
      SLD COMMON ELEMENTS version 1.0.20 (2002-09-21)
    </xsd:documentation>
  </xsd:annotation>

<!-- ********************************************************************** -->
  <xsd:simpleType name="VersionType">
    <xsd:annotation>
      <xsd:documentation>
        The "VersionType" merely restricts the version string that may
        be used with XML documents based on this schema.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="1.0.20"/>
    </xsd:restriction>
  </xsd:simpleType>

<!-- ********************************************************************** -->
  <xsd:element name="Name" type="xsd:string"/>

  <xsd:element name="Description">
    <xsd:annotation>
      <xsd:documentation>
        A "Description" gives human-readable descriptive information for
        the object it is included within.
      </xsd:documentation>
    </xsd:annotation>
```

```
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="Title" type="xsd:string" minOccurs="0"/>
        <xsd:element name="Abstract" type="xsd:string" minOccurs="0"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

<!-- ********************************************************************* -->
  <xsd:element name="FeatureTypeName" type="xsd:string"/>

<!-- ********************************************************************* -->
  <xsd:element name="OnlineResource">
    <xsd:annotation>
      <xsd:documentation>
        An "OnlineResource" is typically used to refer to an HTTP URL.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:attributeGroup ref="xlink:simpleLink"/>
    </xsd:complexType>
  </xsd:element>

<!-- ********************************************************************* -->
  <xsd:element name="InlineContent">
    <xsd:annotation>
      <xsd:documentation>
        "InlineContent" is XML- or base64-encoded encoded content in some
        externally-defined format that is included in an SLD in-line.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType mixed="true">
      <xsd:sequence>
        <xsd:any minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="encoding" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="xml"/>
            <xsd:enumeration value="base64"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

FeatureStyle.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.opengis.net/sld"
            xmlns:sld="http://www.opengis.net/sld"
            xmlns:xlink="http://www.w3.org/1999/xlink"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            elementFormDefault="qualified">
  <xsd:import namespace="http://www.w3.org/1999/xlink"
              schemaLocation="../../gml/2.1/xlinks.xsd"/>

<!-- ********************************************************************* -->
  <xsd:annotation>
    <xsd:documentation>
      SLD COMMON ELEMENTS version 1.0.20 (2002-09-21)
    </xsd:documentation>
  </xsd:annotation>
```

```
<!-- ********************************************************************** -->
  <xsd:simpleType name="VersionType">
    <xsd:annotation>
      <xsd:documentation>
        The "VersionType" merely restricts the version string that may
        be used with XML documents based on this schema.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="1.0.20"/>
    </xsd:restriction>
  </xsd:simpleType>

<!-- ********************************************************************** -->
  <xsd:element name="Name" type="xsd:string"/>

  <xsd:element name="Description">
    <xsd:annotation>
      <xsd:documentation>
        A "Description" gives human-readable descriptive information for
        the object it is included within.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="Title" type="xsd:string" minOccurs="0"/>
        <xsd:element name="Abstract" type="xsd:string" minOccurs="0"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

<!-- ********************************************************************** -->
  <xsd:element name="FeatureTypeName" type="xsd:string"/>

<!-- ********************************************************************** -->
  <xsd:element name="OnlineResource">
    <xsd:annotation>
      <xsd:documentation>
        An "OnlineResource" is typically used to refer to an HTTP URL.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:attributeGroup ref="xlink:simpleLink"/>
    </xsd:complexType>
  </xsd:element>

<!-- ********************************************************************** -->
  <xsd:element name="InlineContent">
    <xsd:annotation>
      <xsd:documentation>
        "InlineContent" is XML- or base64-encoded encoded content in some
        externally-defined format that is included in an SLD in-line.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType mixed="true">
      <xsd:sequence>
        <xsd:any minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="encoding" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="xml"/>
            <xsd:enumeration value="base64"/>
          </xsd:restriction>
```

```
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:complexType>
  </xsd:element>

</xsd:schema>
```

StyledLayerDescriptor.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.opengis.net/sld"
            xmlns:sld="http://www.opengis.net/sld"
            xmlns:ogc="http://www.opengis.net/ogc"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            elementFormDefault="qualified">
  <xsd:include schemaLocation="FeatureStyle.xsd"/>
  <xsd:import namespace="http://www.opengis.net/ogc"
              schemaLocation="../../filter/1.0.0/filter.xsd"/>

<!-- ********************************************************************* -->
  <xsd:annotation>
    <xsd:documentation>
      STYLED LAYER DESCRIPTOR version 1.0.20 (2002-09-21)
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="StyledLayerDescriptor">
    <xsd:annotation>
      <xsd:documentation>
        A StyledLayerDescriptor is a sequence of styled layers, represented
        at the first level by NamedLayer and UserLayer elements.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:Name" minOccurs="0"/>
        <xsd:element ref="sld:Description" minOccurs="0"/>
        <xsd:element ref="sld:EnvironmentVariables" minOccurs="0"/>
        <xsd:element ref="sld:UseSLDLibrary" minOccurs="0"
                                             maxOccurs="unbounded"/>
        <xsd:choice minOccurs="0" maxOccurs="unbounded">
          <xsd:element ref="sld:NamedLayer"/>
          <xsd:element ref="sld:UserLayer"/>
        </xsd:choice>
      </xsd:sequence>
      <xsd:attribute name="version" type="sld:VersionType" use="required"/>
    </xsd:complexType>
  </xsd:element>

<!-- ********************************************************************* -->
  <xsd:annotation>
    <xsd:documentation>
      RUN-TIME ENVIRONMENT VARIABLES
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="EnvironmentVariables">
    <xsd:annotation>
      <xsd:documentation>
        An "EnvironmentVariables" element includes an XML blob which
        can be used to declare the values to use for global environment
        variables during the execution of the styling operations.
        These variables may be refered to in styles and symbols in the
        same way as feature properties.
```

```
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:any/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

<!-- ********************************************************************* -->
  <xsd:annotation>
    <xsd:documentation>
      SLD LIBRARIES
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="UseSLDLibrary">
    <xsd:annotation>
      <xsd:documentation>
        The UseSLDLibrary tag specifies that an external SLD document
        should be used as a "library" of named layers and styles to
        augment the set of named layers and styles that are available
        for use inside of a WMS.  In the event of name collisions, the
        SLD library takes precedence over the ones internal to the WMS.
        Any number of libraries may be specified in an SLD and each
        successive library takes precedence over the former ones in the
        case of name collisions.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:OnlineResource"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

<!-- ********************************************************************* -->
  <xsd:annotation>
    <xsd:documentation>
      LAYERS AND STYLES
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="NamedLayer">
    <xsd:annotation>
      <xsd:documentation>
        A NamedLayer is a layer of data that has a name advertised by a WMS.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:Name"/>
        <xsd:element ref="sld:Description" minOccurs="0"/>
        <xsd:element ref="sld:LayerFeatureConstraints" minOccurs="0"/>
        <xsd:choice minOccurs="0" maxOccurs="unbounded">
          <xsd:element ref="sld:NamedStyle"/>
          <xsd:element ref="sld:UserStyle"/>
        </xsd:choice>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="NamedStyle">
    <xsd:annotation>
      <xsd:documentation>
```

```
      A NamedStyle is used to refer to a style that has a name in a WMS.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="sld:Name"/>
      <xsd:element ref="sld:Description" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:element name="UserLayer">
  <xsd:annotation>
    <xsd:documentation>
      A UserLayer allows a user-defined layer to be built from WFS and
      WCS data.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="sld:Name" minOccurs="0"/>
      <xsd:element ref="sld:Description" minOccurs="0"/>
      <xsd:element ref="sld:RemoteOWS" minOccurs="0"/>
      <xsd:element ref="sld:LayerFeatureConstraints"/>
      <xsd:element ref="sld:UserStyle" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:element name="RemoteOWS">
  <xsd:annotation>
    <xsd:documentation>
      A RemoteOWS gives a reference to a remote WFS/WCS/other-OWS server.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="sld:Service"/>
      <xsd:element ref="sld:OnlineResource"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:element name="Service">
  <xsd:annotation>
    <xsd:documentation>
      A Service refers to the type of a remote OWS server.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="WFS"/>
      <xsd:enumeration value="WCS"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

<xsd:element name="LayerFeatureConstraints">
  <xsd:annotation>
    <xsd:documentation>
      LayerFeatureConstraints define what features &amp;amp; feature types are
      referenced in a layer.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
```

```
      <xsd:sequence>
        <xsd:element ref="sld:FeatureTypeConstraint" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="FeatureTypeConstraint">
    <xsd:annotation>
      <xsd:documentation>
        A FeatureTypeConstraint identifies a specific feature type and
        supplies fitlering.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:FeatureTypeName" minOccurs="0"/>
        <xsd:element ref="ogc:Filter" minOccurs="0"/>
        <xsd:element ref="sld:Extent" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="Extent">
    <xsd:annotation>
      <xsd:documentation>
        An Extent gives feature/coverage/raster/matrix dimension extent.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:Name"/>
        <xsd:element ref="sld:Value"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="Value" type="xsd:string"/>

  <xsd:element name="UserStyle">
    <xsd:annotation>
      <xsd:documentation>
        A UserStyle allows user-defined styling and is semantically
        equivalent to a WMS named style.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:Name" minOccurs="0"/>
        <xsd:element ref="sld:Description" minOccurs="0"/>
        <xsd:element ref="sld:IsDefault" minOccurs="0"/>
        <xsd:element ref="sld:FeatureStyle" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="IsDefault" type="xsd:string"/>

</xsd:schema>
```

Symbol.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.opengis.net/sld"
            xmlns:sld="http://www.opengis.net/sld"
            xmlns:ogc="http://www.opengis.net/ogc"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            elementFormDefault="qualified">
```

```
  <xsd:include schemaLocation="common.xsd"/>
  <xsd:import namespace="http://www.opengis.net/ogc"
              schemaLocation="../../filter/1.0.0/filter.xsd"/>

<!-- ********************************************************************* -->
  <xsd:annotation>
    <xsd:documentation>
      SLD SYMBOL version 1.0.20 (2002-09-21)
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="Symbol" type="sld:SymbolType" abstract="true"/>

  <xsd:complexType name="SymbolType" abstract="true">
    <xsd:annotation>
      <xsd:documentation>
        A "SymbolType" is an abstract type for encoding the graphical
        properties used to portray geographic information.  Concrete symbol
        types are derived from this base type.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element ref="sld:Name" minOccurs="0"/>
      <xsd:element ref="sld:Description" minOccurs="0"/>
      <xsd:element ref="sld:BaseSymbol" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="version" type="sld:VersionType"/>
  </xsd:complexType>

  <xsd:element name="BaseSymbol">
    <xsd:annotation>
      <xsd:documentation>
        A "BaseSymbol" defines the default properties of a symbol to
        be those of an external symbol, which will frequently be inside
        of an OGC symbol(izer) repository.  The symbol properties given
        in-line override the base-symbol properties.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:OnlineResource"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

<!-- ********************************************************************* -->
  <xsd:annotation>
    <xsd:documentation>
      LINE SYMBOL
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="LineSymbol" substitutionGroup="sld:Symbol">
    <xsd:annotation>
      <xsd:documentation>
        A LineSymbol is used to render a "stroke" along a linear geometry.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="sld:SymbolType">
          <xsd:sequence>
            <xsd:element ref="sld:Geometry" minOccurs="0"/>
            <xsd:element ref="sld:Stroke" minOccurs="0"/>
          </xsd:sequence>
```

```
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

<xsd:element name="Geometry">
  <xsd:annotation>
    <xsd:documentation>
      A Geometry gives reference to a (the) geometry property of a
      feature to be used for rendering.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="ogc:PropertyName"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:element name="Stroke">
  <xsd:annotation>
    <xsd:documentation>
      A "Stroke" specifies the appearance of a linear geometry.  It is
      defined in parallel with SVG strokes.  The following SvgParameters
      may be used: "stroke" (color), "stroke-opacity", "stroke-width",
      "stroke-linejoin", "stroke-linecap", "stroke-dasharray", and
      "stroke-dashoffset".  Others are not officially supported.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:choice minOccurs="0">
        <xsd:element ref="sld:GraphicFill"/>
        <xsd:element ref="sld:GraphicStroke"/>
      </xsd:choice>
      <xsd:element ref="sld:SvgParameter" minOccurs="0"
                   maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:element name="SvgParameter">
  <xsd:annotation>
    <xsd:documentation>
      A "SvgParameter" refers to an SVG/CSS graphical-formatting
      parameter.  The parameter is identified using the "name" attribute
      and the content of the element gives the SVG/CSS-coded value.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType mixed="true">
    <xsd:complexContent>
      <xsd:extension base="sld:ParameterValueType">
        <xsd:attribute name="name" type="xsd:string" use="required"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

<xsd:complexType name="ParameterValueType" mixed="true">
  <xsd:annotation>
    <xsd:documentation>
      The "ParameterValueType" uses WFS-Filter expressions to give
      values for SLD graphic parameters.  A "mixed" element-content
      model is used with textual substitution for values.
    </xsd:documentation>
```

```
    </xsd:annotation>
    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:element ref="ogc:expression"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:element name="GraphicFill">
    <xsd:annotation>
      <xsd:documentation>
        A "GraphicFill" defines repeated-graphic filling (stippling)
        pattern for an area geometry.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:Graphic"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="GraphicStroke">
    <xsd:annotation>
      <xsd:documentation>
        A "GraphicStroke" defines a repated-linear graphic pattern to be used
        for stroking a line.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:Graphic"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

<!-- ********************************************************************** -->
  <xsd:annotation>
    <xsd:documentation>
      POLYGON SYMBOL
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="PolygonSymbol" substitutionGroup="sld:Symbol">
    <xsd:annotation>
      <xsd:documentation>
        A "PolygonSymbol" specifies the rendering of a polygon or
        area geometry, including its interior fill and border stroke.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="sld:SymbolType">
          <xsd:sequence>
            <xsd:element ref="sld:Geometry" minOccurs="0"/>
            <xsd:element ref="sld:Fill" minOccurs="0"/>
            <xsd:element ref="sld:Stroke" minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="Fill">
    <xsd:annotation>
      <xsd:documentation>
        A "Fill" specifies the pattern for filling an area geometry.
```

```
        The allowed SvgParameters are: "fill" (color) and "fill-opacity".
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:GraphicFill" minOccurs="0"/>
        <xsd:element ref="sld:SvgParameter" minOccurs="0"
                     maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

<!-- ********************************************************************** -->
  <xsd:annotation>
    <xsd:documentation>
      POINT SYMBOL
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="PointSymbol" substitutionGroup="sld:Symbol">
    <xsd:annotation>
      <xsd:documentation>
        A "PointSymbol" specifies the rendering of a "graphic symbol"
        at a point.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="sld:SymbolType">
          <xsd:sequence>
            <xsd:element ref="sld:Geometry" minOccurs="0"/>
            <xsd:element ref="sld:Graphic" minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="Graphic">
    <xsd:annotation>
      <xsd:documentation>
        A "Graphic" specifies or refers to a "graphic symbol" with inherent
        shape, size, and coloring.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:choice minOccurs="0" maxOccurs="unbounded">
          <xsd:element ref="sld:ExternalGraphic"/>
          <xsd:element ref="sld:Mark"/>
        </xsd:choice>
        <xsd:sequence>
          <xsd:element ref="sld:Opacity" minOccurs="0"/>
          <xsd:element ref="sld:Size" minOccurs="0"/>
          <xsd:element ref="sld:Rotation" minOccurs="0"/>
        </xsd:sequence>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="Opacity" type="sld:ParameterValueType"/>
  <xsd:element name="Size" type="sld:ParameterValueType"/>
  <xsd:element name="Rotation" type="sld:ParameterValueType"/>

  <xsd:element name="ExternalGraphic">
    <xsd:annotation>
```

```
      <xsd:documentation>
        An "ExternalGraphic" gives a reference to an raster or vector
        graphical object, either online or inline, in an externally-defined
        graphic format.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:choice>
          <xsd:element ref="sld:OnlineResource"/>
          <xsd:element ref="sld:InlineContent"/>
        </xsd:choice>
        <xsd:element ref="sld:Format"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="Format" type="xsd:string"/>

  <xsd:element name="Mark">
    <xsd:annotation>
      <xsd:documentation>
        A "Mark" specifies a geometric shape and applies coloring to it.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:WellKnownName" minOccurs="0"/>
        <xsd:element ref="sld:Fill" minOccurs="0"/>
        <xsd:element ref="sld:Stroke" minOccurs="0"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="WellKnownName" type="xsd:string"/>

<!-- ========================================================================= -->
  <xsd:annotation>
    <xsd:documentation>
      TEXT SYMBOL
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="TextSymbol" substitutionGroup="sld:Symbol">
    <xsd:annotation>
      <xsd:documentation>
        A "TextSymbol" is used to render text labels according to
        various graphical parameters.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="sld:SymbolType">
          <xsd:sequence>
            <xsd:element ref="sld:Geometry" minOccurs="0"/>
            <xsd:element ref="sld:Label" minOccurs="0"/>
            <xsd:element ref="sld:Font" minOccurs="0"/>
            <xsd:element ref="sld:LabelPlacement" minOccurs="0"/>
            <xsd:element ref="sld:Halo" minOccurs="0"/>
            <xsd:element ref="sld:Fill" minOccurs="0"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="Label" type="sld:ParameterValueType">
```

```
  <xsd:annotation>
    <xsd:documentation>
      A "Label" specifies the textual content to be rendered.
    </xsd:documentation>
  </xsd:annotation>
</xsd:element>

<xsd:element name="Font">
  <xsd:annotation>
    <xsd:documentation>
      A "Font" element specifies the text font to use.  The allowed
      SvgParameters are: "font-family", "font-style", "font-weight",
      and "font-size".
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="sld:SvgParameter" minOccurs="0"
                   maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:element name="LabelPlacement">
  <xsd:annotation>
    <xsd:documentation>
      The "LabelPlacement" specifies where and how a text label should
      be rendered relative to a geometry.  The present mechanism is
      poorly aligned with CSS/SVG.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:choice>
      <xsd:element ref="sld:PointPlacement"/>
      <xsd:element ref="sld:LinePlacement"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>

<xsd:element name="PointPlacement">
  <xsd:annotation>
    <xsd:documentation>
      A "PointPlacement" specifies how a text label should be rendered
      relative to a geometric point.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="sld:AnchorPoint" minOccurs="0"/>
      <xsd:element ref="sld:Displacement" minOccurs="0"/>
      <xsd:element ref="sld:Rotation" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:element name="AnchorPoint">
  <xsd:annotation>
    <xsd:documentation>
      An "AnchorPoint" identifies the location inside of a text label to
      use an an 'anchor' for positioning it relative to a point geometry.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="sld:AnchorPointX"/>
```

```
        <xsd:element ref="sld:AnchorPointY"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="AnchorPointX" type="sld:ParameterValueType"/>
  <xsd:element name="AnchorPointY" type="sld:ParameterValueType"/>

  <xsd:element name="Displacement">
    <xsd:annotation>
      <xsd:documentation>
        A "Displacement" gives X and Y offset displacements to use for
        rendering a text label near a point.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:DisplacementX"/>
        <xsd:element ref="sld:DisplacementY"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="DisplacementX" type="sld:ParameterValueType"/>
  <xsd:element name="DisplacementY" type="sld:ParameterValueType"/>

  <xsd:element name="LinePlacement">
    <xsd:annotation>
      <xsd:documentation>
        A "LinePlacement" specifies how a text label should be rendered
        relative to a linear geometry.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:PerpendicularOffset" minOccurs="0"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="PerpendicularOffset" type="sld:ParameterValueType">
    <xsd:annotation>
      <xsd:documentation>
        A "PerpendicularOffset" gives the perpendicular distance away
        from a line to draw a label.
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>

  <xsd:element name="Halo">
    <xsd:annotation>
      <xsd:documentation>
        A "Halo" fills an extended area outside the glyphs of a rendered
        text label to make the label easier to read over a background.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:Radius" minOccurs="0"/>
        <xsd:element ref="sld:Fill" minOccurs="0"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="Radius" type="sld:ParameterValueType"/>

<!-- ********************************************************************** -->
  <xsd:annotation>
```

```
  <xsd:documentation>
    RASTER SYMBOL
  </xsd:documentation>
</xsd:annotation>

<xsd:element name="RasterSymbol" substitutionGroup="sld:Symbol">
  <xsd:annotation>
    <xsd:documentation>
      A "RasterSymbol" is used to specify the rendering of
      raster/matrix-coverage data (e.g., satellite images, DEMs).
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="sld:SymbolType">
        <xsd:sequence>
          <xsd:element ref="sld:Geometry" minOccurs="0"/>
          <xsd:element ref="sld:Opacity" minOccurs="0"/>
          <xsd:element ref="sld:ChannelSelection" minOccurs="0"/>
          <xsd:element ref="sld:OverlapBehavior" minOccurs="0"/>
          <xsd:element ref="sld:ColorMap" minOccurs="0"/>
          <xsd:element ref="sld:ContrastEnhancement" minOccurs="0"/>
          <xsd:element ref="sld:ShadedRelief" minOccurs="0"/>
          <xsd:element ref="sld:ImageOutline" minOccurs="0"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>

<xsd:element name="ChannelSelection">
  <xsd:annotation>
    <xsd:documentation>
      "ChannelSelection" specifies the false-color channel selection
      for a multi-spectral raster source.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:choice>
      <xsd:sequence>
        <xsd:element ref="sld:RedChannel"/>
        <xsd:element ref="sld:GreenChannel"/>
        <xsd:element ref="sld:BlueChannel"/>
      </xsd:sequence>
      <xsd:element ref="sld:GrayChannel"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<xsd:element name="RedChannel" type="sld:SelectedChannelType"/>
<xsd:element name="GreenChannel" type="sld:SelectedChannelType"/>
<xsd:element name="BlueChannel" type="sld:SelectedChannelType"/>
<xsd:element name="GrayChannel" type="sld:SelectedChannelType"/>
<xsd:complexType name="SelectedChannelType">
  <xsd:sequence>
    <xsd:element ref="sld:SourceChannelName"/>
    <xsd:element ref="sld:ContrastEnhancement" minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:element name="SourceChannelName" type="xsd:string"/>

<xsd:element name="OverlapBehavior">
  <xsd:annotation>
    <xsd:documentation>
      "OverlapBehavior" tells a system how to behave when multiple
      raster images in a layer overlap each other, for example with
```

```
      satellite-image scenes.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="LATEST_ON_TOP"/>
      <xsd:enumeration value="EARLIEST_ON_TOP"/>
      <xsd:enumeration value="AVERAGE"/>
      <xsd:enumeration value="RANDOM"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

<xsd:element name="ColorMap">
  <xsd:annotation>
    <xsd:documentation>
      A "ColorMap" defines either the colors of a pallet-type raster
      source or the mapping of numeric pixel values to colors.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:choice minOccurs="0" maxOccurs="unbounded">
      <xsd:element ref="sld:ColorMapEntry"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
<xsd:element name="ColorMapEntry">
  <xsd:complexType>
    <xsd:attribute name="color" type="xsd:string" use="required"/>
    <xsd:attribute name="opacity" type="xsd:double"/>
    <xsd:attribute name="quantity" type="xsd:double"/>
    <xsd:attribute name="label" type="xsd:string"/>
  </xsd:complexType>
</xsd:element>

<xsd:element name="ContrastEnhancement">
  <xsd:annotation>
    <xsd:documentation>
      "ContrastEnhancement" defines the 'stretching' of contrast for a
      channel of a false-color image or for a whole grey/color image.
      Contrast enhancement is used to make ground features in images
      more visible.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:choice minOccurs="0">
        <xsd:element ref="sld:Normalize"/>
        <xsd:element ref="sld:Histogram"/>
      </xsd:choice>
      <xsd:element ref="sld:GammaValue" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element name="Normalize">
  <xsd:complexType/>
</xsd:element>
<xsd:element name="Histogram">
  <xsd:complexType/>
</xsd:element>
<xsd:element name="GammaValue" type="xsd:double"/>

<xsd:element name="ShadedRelief">
  <xsd:annotation>
    <xsd:documentation>
```

```
        "ShadedRelief" specifies the application of relief shading
        (or "hill shading") to a DEM raster to give it somewhat of a
        three-dimensional effect and to make elevation changes more
        visible.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="sld:BrightnessOnly" minOccurs="0"/>
        <xsd:element ref="sld:ReliefFactor" minOccurs="0"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="BrightnessOnly" type="xsd:boolean"/>
  <xsd:element name="ReliefFactor" type="xsd:double"/>

  <xsd:element name="ImageOutline">
    <xsd:annotation>
      <xsd:documentation>
        "ImageOutline" specifies how individual source rasters in
        a multi-raster set (such as a set of satellite-image scenes)
        should be outlined to make the individual-image locations visible.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:choice>
        <xsd:element ref="sld:LineSymbol"/>
        <xsd:element ref="sld:PolygonSymbol"/>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>

<!-- ********************************************************************** -->
  <xsd:annotation>
    <xsd:documentation>
      MAPPED-COLOR SYMBOL (Choropleth)
    </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="MappedColorSymbol" substitutionGroup="sld:Symbol">
    <xsd:annotation>
      <xsd:documentation>
        A "MappedColorSymbol" is used to specify the mapping of colors to
        ranges of values of a control variable/expression.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:complexContent>
        <xsd:extension base="sld:SymbolType">
          <xsd:sequence>
            <xsd:element ref="sld:LookupValue"/>
            <xsd:element ref="sld:ColorMap"/>
            <xsd:element ref="sld:MappedColorSubSymbol"/>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="LookupValue" type="sld:ParameterValueType"/>

  <xsd:element name="MappedColorSubSymbol">
    <xsd:annotation>
      <xsd:documentation>
        The MappedColorSubSymbol identifies the the symbol to which
        to apply computed colors.  The computed color is used as the
```

```
        'default color' in the sub-symbol.  The "matchTo" attribute tells
        whether to match the lookup value to the "quantity" or "label"
        attributes of the 'choropleth' ColorMap.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:choice>
        <xsd:element ref="sld:LineSymbol"/>
        <xsd:element ref="sld:PolygonSymbol"/>
        <xsd:element ref="sld:PointSymbol"/>
        <xsd:element ref="sld:TextSymbol"/>
        <xsd:element ref="sld:RasterSymbol"/>
      </xsd:choice>
      <xsd:attribute name="matchTo" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="quantity"/>
            <xsd:enumeration value="label"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:complexType>
  </xsd:element>

</xsd:schema>
```

Annex B
(informative)

Which styles correspond to which element — Quick reference guide

Element		Style name	Next Style name
Any element of text for which the correct style to use is not known with certainty or for which no other style of the template is suitable		Special	Normal
Annex	heading, status and title	ANNEX	Normal
Bibliography	reference entry	bibliography	bibliography
	title	zzBiblio	bibliography
Clause	annexes	a2	Normal
	body of document	Heading 1	Normal
Code listing		CODE	CODE
Indented code listing	no indentation	Code 1	Code 1
	indented one level (0.25”)	Code 2	Code 2
	indented two levels (0.50”)	Code 3	Code 3
	indented 3 levels (0.75”)	Code 4	Code 4
	indented 4 levels (1.00”)	Code 5	Code 5
	indented 5 levels (1.25”)	Code 6	Code 6
	indented 6 levels (1.50”)	Code 7	Code 7
	indented 7 levels (1.75”)	Code 8	Code 8
	indented 8 levels (2.00”)	Code 9	Code 9
	indented 9 levels (2.25”)	Code 10	Code 10
	indented 10 levels (2.50”)	Code 11	Code 11
Definition		Definition	TermNum
Displayed mathematical and chemical formulae		Formula	Normal
Example		Example	Normal
Figure	figure artwork	Figure art	Figure title
	note	Note	Normal
	footnote	Figure footnote	Figure footnote
	title	Figure title	Normal
Figure footnote		Figure footnote	Figure footnote
Footnote	reference	Footnote Reference	(none)
	text	Footnote Text	Footnote Text

Element			Style name	Next Style name
Footnote text			Footnote Text	Footnote Text
Foreword	text		Foreword	Foreword
	title		ZzForeword	Normal
Index	entry		Index 1	Normal
	entry heading		Index Heading	Index 1
	«Index» title		ZzIndex	Normal
Introduction	text		Normal	Normal
	title		Introduction	Normal
List	ordered	level 1	List Number	List Number
		level 2	List Number 2	List Number 2
		level 3	List Number 3	List Number 3
		level 4	List Number 4	List Number 4
	unordered	level 1	List Continue	List Continue
		level 2	List Continue 2	List Continue 2
		level 3	List Continue 3	List Continue 3
		level 4	List Continue 4	List Continue 4
Normative reference	cross-reference to a normative reference defined in the «Normative references» clause		ExtXref	(none)
	in «Normative references» clause		RefNorm	Normal
Note integrated in text, figures and tables			Note	Normal
Paragraph			Normal	Normal
Subclause without title	level 1		p2	Normal
	level 2		p3	Normal
	level 3		p4	Normal
	level 4		p5	Normal
	level 5		p6	Normal
Subclause with title	annexes	level 1	a 2	Normal
		level 2	a 3	Normal
		level 3	a 4	Normal
		level 4	a 5	Normal
	body of text	level 1	Heading 2	Normal
		level 2	Heading 3	Normal
		level 3	Heading 4	Normal
		level 4	Heading 5	Normal
		level 5	Heading 6	Normal

Element			Style name	Next Style name
Table	title		Table title	Normal
	table cell		Body Text	Body Text
	table cell of a big table		Body Text 2	Body Text 2
	table cell of a very big table		Body Text 3	Body Text 3
	note		Note	Normal
	table footnote		See Table footnote.	See Table footnote.
Table footnote	identification letter of table footnote		TableFootNoteXref	(none)
	reference		TableFootNoteXref	(none)
	text		Table footnote	Table footnote
Table of contents	«Contents» title		ZzContents	TOC 1
	entries	level 1	TOC 1	Normal
		level 2	TOC 2	Normal
		level 3	TOC 3	Normal
		level 4	TOC 4	Normal
		level 5	TOC 5	Normal
		level 6	TOC 6	Normal
		unnumbered elements	TOC 9	Normal
Term	cross-reference to a term defined in the «Terms and definitions» clause		Defterms	(none)
	in the «Terms and definitions» clause in a terminology document:			
	reference number for term		TermNum	Term(s)
	term name		Term(s)	Definition
	definition of term		Definition	TermNum

Annex C – Discussion of Metadata Documents

This annex contains a discussion of the metadata documents that are used to describe both styles and symbols. Section C.1 walks through a sample document. Section C.2 describes issues with the metadata. Section C.3 gives a listing of the document that was explained in section C.1. Section C.4 gives the XML schema instance that describes style and symbol metadata instances.

C.1 Style and Symbol Metadata (informative)

This section describes the style and symbol metadata schema by stepping through a fictitious metadata instance. The contents of this section are informative only.

First, there is the standard XML header:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Then the opening tag:

```
<StyleMetadata xmlns="http://www.opengis.org/sms">
```

When the metadata is describing a style object, the opening tag will be `StyleMetadata`. If the metadata describes a symbol, the opening tag will be `SymbolMetadata`. Both `Style-` and `SymbolMetadata` have the exact same structure.

All content of the `StyleMetadata` tag is optional except for the `ContentURL` which indicates where the style object can be retrieved. This gives a URL where the style (or symbol) described by this metadata can be retrieved:

```
<ContentURL>http://www.polexis.com/sms/repository.pl?REQUES
T=GetObject&amp;ID=1234567890</ContentURL>
```

Note that in this example, the URL is actually a request to a repository to retrieve an item based on its ID. This might happen if the style object were first inserted into a Web Object Service before being registered with a registry.

In order to identify style or symbol instances through the metadata, a human readable name is associated with a style or symbol instance.

<Name>VMAP0 Oceans</Name>

The name value is used by clients to construct a list of appropriate style or symbol names when multiple style or symbol metadata documents are returned from a registry query.

The next optional tag is Description. This content allows the creator of style or symbol metadata to provide users an informative description about the style or symbol instance in which the metadata refers.

<Description>NIMA VMAP0 Style for Oceans or Sea Boundaries</Description>

Next comes a tag which gives the two character code indicating the language used in any comments or descriptions contained in the metadata:

```
<Language>EN</Language>
```

The content of this tag must be a two character language code as defined by ISO 639-2. (The valid possibilities are also listed in Annex F of ISO 19115.)

Next is a tag that gives contact information for the author or maintainer of the metadata and style. This tag is a subset of the information present in the ISO 19115 data type "`CI_ResponsibleParty`".

```
<Contact>
    <IndividualName>Chris Dillard</IndividualName>
    <OrganisationName>Polexis, Inc.</OrganisationName>
    <PositionName>Software Engineer</PositionName>
    <ContactInfo>
        <PhoneNumber>(555) 555-5555</PhoneNumber>
        <Address>123 A St., Nowhere, CA 95000</Address>

<OnlineResource>http://www.polexis.com</OnlineResource>
        <ContactInstructions>
            Either call the given number or e-mail
            cdillard@polexis.com
        </ContactInstructions>
    </ContactInfo>
</Contact>
```

This is followed by a tag that gives the time when the style (or symbol) was last modified:

```
<DateStamp>2002-07-22T14:30:00-08:00</DateStamp>
```

Next is a tag whose content can list the schema for the style or symbol document described by this metadata:

```
<ApplicationSchemaInfo>
    <Name>SLD 0.7.3</Name>

<SchemaLocation>http://www.abc.def/ghi.xsd</SchemaLocation>
    <SchemaLanguage>XMLSchema</SchemaLanguage>
</ApplicationSchemaInfo>
```

The `Name` tag is descriptive, and used only to provide a human readable name for the schema that constrains the style. The content of the `SchemaLocation` tag must be an absolute URL that can be used to retrieve a document (in this case an XML schema instance). And the `SchemaLanguage` tag must contain either "DTD" or "XMLSchema". (Other tags may be introduced in the future to support other schema languages.)

Next is a tag containing a list of feature types:

```
<FeatureTypeList>
    <FeatureType>OCEANSEA_1M</FeatureType>
</FeatureTypeList>
```

Each `FeatureType` element names a data type that is used by the style or symbol or its constituent parts. In the above example, `OCEANSEA_1M` is an implicit reference to an application schema that defines the data type `OCEANSEA_1M`. There is currently not a way to explicitly reference the schema where this type is defined. This may be added in the (near) future.

Next is a tag containing a list of semantic type identifiers.

```
<SemanticTypeIdentifierList>
   <SemanticTypeIdentifier>nima:vmap0:bnd:oceansea
   </SemanticTypeIdentifier>
   <SemanticTypeIdentifier>generic:polygon
   </SemanticTypeIdentifier>
</SemanticTypeIdentifierList>
```

The SemanticTypeIdentifier element describes the suitable feature types in which a feature styles can be used and corresponds to the same SemanticTypeIdentifier element in

the SLD specification. These values can be generic geometries such as generic:point, generic:line, generic:polygon, generic:text and generic:raster when a style can be used with many different feature types.

Next is a tag that contains a list of "library names". These names are human readable strings that describe a collection of styles or symbols to which a style or symbol belongs.

```
<LibraryNameList>
    <LibraryName>NIMA VPF</LibraryName>
</LibraryNameList>
```

The library name may serve as a way to implicitly group styles or symbols. The user may, for example, request all styles in the "NIMA VPF" library. However, there is no standard (yet) that defines the range of valid library names, so it is not clear if this will be usable in an interoperable manner.

In order to support the idea of a hierarchical classification scheme (as defined in ebRIM), there is a `ClassificationList` element that lists references to classifications in well-known schemes. For example:

```
<ClassificationList>
    <Classification scheme="FACC">
        A001
    </Classification>
    <Classification scheme="ApplicationDomain">
        Agriculture
    </Classification>
</ClassificationList>
```

This example snippet says that the style refers to a feature of type "A001" in the FACC classification scheme and that it should be used for styling agricultural features. It has not yet been decided exactly how the pair of strings for the classification scheme will refer to an ebRIM classification node, but it is presumed that an SMS will be able to use the data in a `Classification` node to uniquely identify an ebRIM classification object.

Next there is a `FeaturePropertyList` element that lists which GML properties of the feature will be used by the Rules which comprise the style or symbol.

```
<FeaturePropertyList>
    <FeatureProperty>NUMBER_OF_LANES</FeatureProperty>
</FeaturePropertyList>
```

Currently, this list is only descriptive since there is not a way to explicitly indicate the XPath and/or namespace that the given feature property refers to. As more research is done to determine the best way to reference a feature's properties, the nature of this element may change.

Next in the metadata is an element that lists the types of symbols that comprise the style or symbol to which the metadata refers.

```
<SymbolTypeList>
    <SymbolType>POINT</SymbolType>
    <SymbolType>TEXT</SymbolType>
</SymbolTypeList>
```

This example could describe a style containing two symbols, one that draws a point symbol and one that draws a text label.

The valid choices for the `SymbolType` element are `POINT`, `TEXT`, `LINE`, `POLYGON`, and `RASTER`. If the metadata is describing a symbol, then it must be that case that there is exactly one `SymbolType` child of `SymbolTypeList`.

It was recognized that some styles may depend on external parameters, such as the time of day or the brightness of the location containing a computer screen. In order to support this, the style and symbol metadata have the following element which lists the names of parameters to the style or symbol:

```
<StyleParameterList>
    <StyleParameter>TIME_OF_DAY</StyleParameter>
</StyleParameterList>
```

No mechanism exists (yet) to specify the data type that the given property is expected to contain. Also, it is not yet defined how the style or symbol elements (in SLD 0.7.3) can reference these properties. More work has to be done to clarify these points.

Next, there is a section in the metadata for arbitrary keywords that the user wishes to list:

```
<KeywordList>
        <Keyword>sample</Keyword>
        <Keyword>example</Keyword>
</KeywordList>
```

This is followed by the closing tag that ends the metadata document:

```
</StyleMetadata>
```

C.2 Unresolved Issues (informative)

The issues mentioned in this section C.1 are collected here for easier reference:

- The `FeatureTypeList` tag may need a way to specify the schema of the feature types that are listed (although it is unclear how an SMS client would make use of this, other than to provide feedback to the human user).
- The `LibraryNameList` tag may be too flexible to be useful in an interoperable manner.
- It is unclear how the `ClassificationList` should reference ebRIM classification nodes. Should the classification scheme be a reference (perhaps a URL) to a scheme registered in an OGC registry? Should it just be a "well-known" name?
- It has yet to be decided how the properties listed in `FeaturePropertyList` will reference the GML structure, if at all. Should this be an XPath expression? Should it convey any type information about the feature property?
- The SLD specification does not yet provide a mechanism for styles and symbols to reference the external parameters listed in `StyleParameterList`. (This is not difficult to correct, however, and a couple of mechanisms have been considered.)

C.3 Complete Style Metadata Example (informative)

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- Gives the metadata for a fictitious style that resides in an SMS. -->
<StyleMetadata xmlns="http://www.opengis.org/sms">

    <!-- Gives the URL that the style this metadata describes can be retrieved from. -->

<ContentURL>http://www.polexis.com/sms/repository.pl?REQUEST=GetObject&amp;ID=1234567890<
/ContentURL>

    <!-- Gives a human readable name for the style instance refenced by the metadata. -->
    <Name>VAMAP0 Oceans</Name>

    <!-- Gives an informative description about the style instance refenced by the
metadata. -->
    <Description>NIMA VMAP0 Style for Oceans or Seas</Description


    <!-- Gives the language that comments are written in. -->
    <Language>EN</Language>

    <!-- Gives information about the author of the style. -->
    <Contact>
        <IndividualName>Chris Dillard</IndividualName>
        <OrganisationName>Polexis, Inc.</OrganisationName>
        <PositionName>Software Engineer</PositionName>
        <ContactInfo>
            <PhoneNumber>(619) 542-7230</PhoneNumber>
            <Address>2815 Camino del Rio South, San Diego CA 92108</Address>
            <OnlineResource>http://www.polexis.com</OnlineResource>
            <ContactInstructions>Either call the given number or e-mail
cdillard@polexis.com</ContactInstructions>
        </ContactInfo>
    </Contact>

    <!-- Gives the last time the style was edited. -->
    <DateStamp>2002-07-22T14:30:00-08:00</DateStamp>

    <!-- Gives schema information about the style. -->
    <ApplicationSchemaInfo>
        <Name>SLD 0.7.3</Name>
        <SchemaLocation>http://www.abc.def/ghi.xsd</SchemaLocation>
        <SchemaLanguage>XMLSchema</SchemaLanguage>
    </ApplicationSchemaInfo>

    <!-- Gives the list of feature types to which the style can be applied. -->
    <FeatureTypeList>
        <FeatureType>OCEANSEA_1M</FeatureType>
    </FeatureTypeList>

    <!-- Gives the list of Semantic Types to which the style can be applied. -->
    <SemanticTypeIdentifierList>
         <SemanticTypeIdentifier>nima:vmap0:bnd:oceansea</SemanticTypeIdentifier>
         <SemanticTypeIdentifier>generic:polygon</SemanticTypeIdentifier>
    </SemanticTypeIdentifierList>


    <!-- Gives the name of a library of styles or symbols to which this one belongs.
Examples are: "NIMA VPF", "GeoSym", "MIL-STD-2525B", "USGS Quad Series", etc. -->
    <LibraryNameList>
```

```
        <LibraryName>NIMA VPF</LibraryName>
    </LibraryNameList>

    <!-- Give some classification nodes. -->
    <!-- For now, I guess these are just "well-known" schemes and their corresponding
nodes. -->
    <ClassificationList>
        <Classification scheme="FACC">A001</Classification>
        <Classification scheme="ApplicationDomain">Agriculture</Classification>
    </ClassificationList>

    <!-- Lists which properties of the feature are used by rules in this style. -->
    <FeaturePropertyList>
        <FeatureProperty>NUMBER_OF_LANES</FeatureProperty>
    </FeaturePropertyList>

    <!-- Lists the types of symbols (previously known as symbolizers) contained within
this style or symbol. -->
    <SymbolTypeList>
        <SymbolType>POINT</SymbolType>
        <SymbolType>TEXT</SymbolType>
    </SymbolTypeList>

    <!-- Lists the "external" parameters that can be given to this style or symbol.  This
is designed to handle things like a style that changes depending on the time of day.
This probably needs some more thinking... -->
    <StyleParameterList>
        <StyleParameter>TIME_OF_DAY</StyleParameter>
    </StyleParameterList>

    <!-- Give some user-defined keywords. -->
    <KeywordList>
        <Keyword>sample</Keyword>
        <Keyword>example</Keyword>
    </KeywordList>
</StyleMetadata>
```

C.4 Style and Symbol Metadata Schema (normative)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.opengis.org/sms"
xmlns:tns="http://www.opengis.org/sms" xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
    <!-- Top level elements. -->
    <xs:element name="StyleMetadata" type="tns:StyleOrSymbolMetadataType">
        <xs:annotation>
            <xs:documentation>Gives metadata about a style that is stored in an OGC
registry.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="SymbolMetadata" type="tns:StyleOrSymbolMetadataType">
        <xs:annotation>
            <xs:documentation>Gives metadata about a symbol that is stored in an OGC
registry.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <!-- Subordinate elements. -->
    <xs:element name="Name" type="xs:string">
```

```
        <xs:annotation>
            <xs:documentation>Gives a short, mnemonic name for the style or symbol to
which this metadata refers.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="Description" type="xs:string">
        <xs:annotation>
            <xs:documentation>Gives a brief, free-text description of the style or symbol
to which this metadata refers.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="Language" type="xs:language">
        <xs:annotation>
            <xs:documentation>Gives the two letter code for the language that this
metadata's free text is written in.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="Contact" type="tns:ContactType">
        <xs:annotation>
            <xs:documentation>Gives contact information about the author or maintainer of
this metadata.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="ContactInfo" type="tns:ContactInfoType">
        <xs:annotation>
            <xs:documentation>Information about how to contact the maintainer or
publisher of this metadata.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="ApplicationSchemaInfo" type="tns:ApplicationSchemaInfoType">
        <xs:annotation>
            <xs:documentation>Indicates the schema for the style or symbol that this
metadata describes.</xs:documentation>
        </xs:annotation>
    </xs:element>
    <!-- Type definitions. -->
    <xs:complexType name="StyleOrSymbolMetadataType">
        <xs:sequence>
            <xs:element name="ContentURL" type="xs:anyURI">
                <xs:annotation>
                    <xs:documentation>This tag gives the URL of the style or symbol that
this metadata refers to.  The URL may point to any server anywhere, not just the server
where the style's existence is registered.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element ref="tns:Name" minOccurs="0"/>
            <xs:element ref="tns:Description" minOccurs="0"/>
            <xs:element ref="tns:Language" minOccurs="0"/>
            <xs:element ref="tns:Contact" minOccurs="0"/>
            <xs:element name="DateStamp" type="xs:dateTime" minOccurs="0">
                <xs:annotation>
                    <xs:documentation>Date and time when this metadata was last
modified.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element ref="tns:ApplicationSchemaInfo" minOccurs="0"/>
            <xs:element name="FeatureTypeList" type="tns:FeatureTypeListType"
minOccurs="0">
                <xs:annotation>
                    <xs:documentation>List of feature types that this style applies
to.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="SemanticTypeIdentifierList"
type="tns:SemanticTypeIdentifierListType" minOccurs="0">
```

```
                <xs:annotation>
                    <xs:documentation>List of semantic types styled by this style or
symbol.  (See the SLD specification for a description of the meaning of "semantic
type".)</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="LibraryNameList" type="tns:LibraryNameListType"
minOccurs="0">
                <xs:annotation>
                    <xs:documentation>List of names of the symbol or style libraries that
an object belongs to.  This will have children whose text is "NIMA VPF" or "MIL-STD-2525"
or similar.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="ClassificationList" type="tns:ClassificationListType"
minOccurs="0">
                <xs:annotation>
                    <xs:documentation>Lists the classifications of this style or
symbol.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="FeaturePropertyList" type="tns:FeaturePropertyListType"
minOccurs="0">
                <xs:annotation>
                    <xs:documentation>Lists the names of the feature properties that this
symbol or style (or its constituent parts) use in their "Rule"
elements.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="SymbolTypeList" type="tns:SymbolTypeListType"
minOccurs="0">
                <xs:annotation>
                    <xs:documentation>Lists the types of symbols that this style (or
symbol) contains.  This list will contain strings such as "POINT", "LINE", and
"POLYGON".</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="StyleParameterList" type="tns:StyleParameterListType"
minOccurs="0">
                <xs:annotation>
                    <xs:documentation>Lists the "external" parameters that this style
depends on.  This may be used, for example, if a style has a dependency on the time of
day.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="KeywordList" type="tns:KeywordListType" minOccurs="0">
                <xs:annotation>
                    <xs:documentation>List of arbitrary keywords that the author of the
style may wish to list in order to faciilitate searching.</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ContactType">
        <xs:sequence>
            <xs:element name="IndividualName" type="xs:string" minOccurs="0">
                <xs:annotation>
                    <xs:documentation>Name(s) of the individual(s) who maintains or
published this metadata.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="OrganisationName" type="xs:string" minOccurs="0">
                <xs:annotation>
                    <xs:documentation>Name of the organisation that maintains or
published this metadata.</xs:documentation>
```

```
                </xs:annotation>
            </xs:element>
            <xs:element name="PositionName" type="xs:string" minOccurs="0">
                <xs:annotation>
                    <xs:documentation>Position of the maintainer(s) of this
metadata.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element ref="tns:ContactInfo" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ContactInfoType">
        <xs:sequence>
            <xs:element name="PhoneNumber" type="xs:string" minOccurs="0"/>
            <xs:element name="Address" type="xs:string" minOccurs="0"/>
            <xs:element name="OnlineResource" type="xs:anyURI" minOccurs="0"/>
            <xs:element name="ContactInstructions" type="xs:string" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ApplicationSchemaInfoType">
        <xs:sequence>
            <xs:element name="Name" type="xs:string" minOccurs="0">
                <xs:annotation>
                    <xs:documentation>Gives a human-readable name of the schema to which
this element refers.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="SchemaLocation" type="xs:anyURI">
                <xs:annotation>
                    <xs:documentation>Gives a URL that can be used to retrieve the schema
for the style or symbol that this metadata refers to.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="SchemaLanguage" type="xs:string">
                <xs:annotation>
                    <xs:documentation>Gives a string that indicates what language the
schema is written in.  It is expected that this will only ever be "DTD" or
"XMLSchema".</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="FeatureTypeListType">
        <xs:sequence>
            <xs:element name="FeatureType" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="SemanticTypeIdentifierListType">
        <xs:sequence>
            <xs:element name="SemanticTypeIdentifier" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="LibraryNameListType">
        <xs:sequence>
            <xs:element name="LibraryName" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ClassificationListType">
        <xs:sequence>
            <xs:element name="Classification" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:simpleContent>
```

```
                        <xs:extension base="xs:string">
                            <xs:attribute name="scheme" type="xs:string"
use="required"/>
                        </xs:extension>
                    </xs:simpleContent>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="FeaturePropertyListType">
        <xs:sequence>
            <xs:element name="FeatureProperty" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="SymbolTypeListType">
        <xs:sequence>
            <xs:element name="SymbolType" minOccurs="0" maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>This should probably have an enumerated list of
valid values.</xs:documentation>
                </xs:annotation>
                <xs:simpleType>
                    <xs:restriction base="xs:string"/>
                </xs:simpleType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="StyleParameterListType">
        <xs:sequence>
            <xs:element name="StyleParameter" type="xs:string" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="KeywordListType">
        <xs:sequence>
            <xs:element name="Keyword" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

Annex D - Example of an SLD document using an SMS

(normative)

D.1 Main SLD Document with object IDs for styles in the style repository.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!-- uses styles & symbols stored in the SMS repository -->

<StyledLayerDescriptor version="1.0.20"
xmlns="http://www.opengis.net/sld"
   xmlns:ogc="http://www.opengis.net/ogc"
   xmlns:xlink="http://www.w3.org/1999/xlink"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.opengis.net/sld
http://schemas.cubewerx.com/schemas/sld/1.0.20/StyledLayerDescriptor.xsd">

   <NamedLayer>
      <Name>OCEANSEA_1M:Foundation</Name>
      <UserStyle>
         <Name>GeoSym</Name>
         <IsDefault>1</IsDefault>
         <FeatureStyle>
            <OnlineResource xlink:type="simple"
               xlink:href="http://demo.cubewerx.com/ows12/wos/cwwos.cgi?oid=STYLES.44"/>
         </FeatureStyle>
      </UserStyle>
   </NamedLayer>

   <NamedLayer>
      <Name>POLBNDA_1M:Foundation</Name>
      <UserStyle>
         <Name>GeoSym</Name>
         <IsDefault>1</IsDefault>
         <FeatureStyle>
            <OnlineResource xlink:type="simple"
               xlink:href="http://demo.cubewerx.com/ows12/wos/cwwos.cgi?oid=STYLES.45"/>
         </FeatureStyle>
      </UserStyle>
   </NamedLayer>

   <NamedLayer>
      <Name>INWATERA_1M:Foundation</Name>
      <UserStyle>
         <Name>GeoSym</Name>
         <IsDefault>1</IsDefault>
         <FeatureStyle>
            <OnlineResource xlink:type="simple"
               xlink:href="http://demo.cubewerx.com/ows12/wos/cwwos.cgi?oid=STYLES.43"/>
         </FeatureStyle>
      </UserStyle>
   </NamedLayer>

   <NamedLayer>
      <Name>SWAMPA_1M:Foundation</Name>
      <UserStyle>
         <Name>GeoSym</Name>
```

```
         <IsDefault>1</IsDefault>
         <FeatureStyle>
            <OnlineResource xlink:type="simple"
               xlink:href="http://demo.cubewerx.com/ows12/wos/cwwos.cgi?oid=STYLES.47"/>
         </FeatureStyle>
      </UserStyle>
   </NamedLayer>

   <NamedLayer>
      <Name>TREESA_1M:Foundation</Name>
      <UserStyle>
         <Name>GeoSym</Name>
         <IsDefault>1</IsDefault>
         <FeatureStyle>
            <OnlineResource xlink:type="simple"
               xlink:href="http://demo.cubewerx.com/ows12/wos/cwwos.cgi?oid=STYLES.48"/>
         </FeatureStyle>
      </UserStyle>
   </NamedLayer>

   <NamedLayer>
      <Name>BUILTUPA_1M:Foundation</Name>
      <UserStyle>
         <Name>GeoSym</Name>
         <IsDefault>1</IsDefault>
         <FeatureStyle>
            <OnlineResource xlink:type="simple"
               xlink:href="http://demo.cubewerx.com/ows12/wos/cwwos.cgi?oid=STYLES.41"/>
         </FeatureStyle>
      </UserStyle>
   </NamedLayer>

   <NamedLayer>
      <Name>COASTL_1M:Foundation</Name>
      <UserStyle>
         <Name>GeoSym</Name>
         <IsDefault>1</IsDefault>
         <FeatureStyle>
            <OnlineResource xlink:type="simple"
               xlink:href="http://demo.cubewerx.com/ows12/wos/cwwos.cgi?oid=STYLES.42"/>
         </FeatureStyle>
      </UserStyle>
   </NamedLayer>

   <NamedLayer>
      <Name>POLBNDL_1M:Foundation</Name>
      <UserStyle>
         <Name>GeoSym</Name>
         <IsDefault>1</IsDefault>
         <FeatureStyle>
            <OnlineResource xlink:type="simple"
               xlink:href="http://demo.cubewerx.com/ows12/wos/cwwos.cgi?oid=STYLES.46"/>
         </FeatureStyle>
      </UserStyle>
   </NamedLayer>

</StyledLayerDescriptor>
```

D.2 Style document (example snippets) from style repository with object IDs for symbols in symbol repository

```
<FeatureStyle version="1.0.20"
xmlns="http://www.opengis.net/sld"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sld
http://schemas.cubewerx.com/schemas/sld/1.0.20/FeatureStyle.xsd">
   <Name>geosym-vmap0-hydro-inwatera</Name>
   <Description>
      <Title>Inland Water Areas</Title>
   </Description>
   <SemanticTypeIdentifier>nima:vmap0:hydro:inwatera</SemanticTypeIdentifier>
   <Rule>
      <Name>BH000</Name>
      <Description>
         <Title>Inland Water</Title>
      </Description>
      <ogc:Filter>
         <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>F_CODE</ogc:PropertyName>
            <ogc:Literal>BH000</ogc:Literal>
         </ogc:PropertyIsEqualTo>
      </ogc:Filter>
      <PolygonSymbol>
         <BaseSymbol>
            <OnlineResource xlink:type="simple"
xlink:href="http://demo.cubewerx.com/ows12/wos/cwwos.cgi?oid=SYMBOLS.54"/>
         </BaseSymbol>
      </PolygonSymbol>
   </Rule>
   <Rule>
      <Name>BH090</Name>
      <Description>
         <Title>Land Subject to Inundation</Title>
      </Description>
      <ogc:Filter>
         <ogc:PropertyIsEqualTo>
            <ogc:PropertyName>F_CODE</ogc:PropertyName>
            <ogc:Literal>BH090</ogc:Literal>
         </ogc:PropertyIsEqualTo>
      </ogc:Filter>
      <PolygonSymbol>
         <BaseSymbol>
            <OnlineResource xlink:type="simple"
xlink:href="http://demo.cubewerx.com/ows12/wos/cwwos.cgi?oid=SYMBOLS.51"/>
         </BaseSymbol>
      </PolygonSymbol>
   </Rule>
</FeatureStyle>
```

D.3 Symbol document (Symbol.51) from a symbol repository with inline graphic.

```
<PolygonSymbol version="1.0.20"
xmlns="http://www.opengis.net/sld"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xlink="http://www.w3.org/1999/xlink"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/sld
http://schemas.cubewerx.com/schemas/sld/1.0.20/Symbol.xsd">
   <Name>GeoSym-4078</Name>
   <Description>
      <Title>Land Subject to Inundation</Title>
   </Description>
   <Fill>
      <GraphicFill>
         <Graphic>
            <ExternalGraphic>
               <InlineContent
encoding="base64">iVBORw0KGgoAAAANSUhEUgAAADwAAAAdAgMAAACLYudyAAAACVBMVEX///+xyOpjktapNoH
L
      AAAAAXRSTlMAQObYZgAAAAFiS0dEAmYLfGQAAAAWdEVYdFNvZnR3YXJlAGdpZjJwbmcgMi40
      LjakM4MXAAAAMUlEQVR42mNg0FrFwLhqBQPTqgUMXKsaGGgNgFasWgm0bQXQtgUMWjS3j97+
      G2D/AgDOGxPtudIE/gAAAABJRU5ErkJggg==</InlineContent>
               <Format>image/png</Format>
            </ExternalGraphic>
         </Graphic>
      </GraphicFill>
   </Fill>
</PolygonSymbol>
```

Bibliography

[1] ISO 31 (all parts), *Quantities and units*.

[2] IEC 60027 (all parts), *Letter symbols to be used in electrical technology*.

[3] ISO 1000, *SI units and recommendations for the use of their multiples and of certain other units*.

[4] OGC 00-014r1 Guidelines for Successful OGC Interface Specifications.

[5] Style Management Service Statement of Requirements (unnumbered).

[6] OGC 02-070 Styled Layer Descriptor Implementation Specification.

[7] OGC 02-058 Web Feature Server Implementation Specification.

[8] OGC 02-059 Filter Encoding Implementation Specification.

[9] OGC 01-068r3 Web Map Service Implementation Specification.

[10] OGC 02-017 Web Map Service HTTP-POST Requests IPR.

[11] OGC 02-050r5 Registry Service Interoperability Program Report.

[12] OGC 02-049 Web Object Service Implementation Specification.

[13] OGC 02-112r2, OGC Abstract Specification Topic 12: Service Architecture, available at: http://www.opengis.org/techno/abstract/02-112.pdf.