Draft

Consistent Symbology For The National Map: The Introduction of Styled Layer Descriptors

John Murphey
Catalog Support Team

Mid-Continent Mapping Center
December 2004

This document is for internal use by USGS Geospatial Information Office personnel. It has not been reviewed for conformance with USGS editorial standards and has not been approved for formal publication. Any us of trade names is for descriptive purposes only and does not constitute endorsement by the US Government.

Fine-grained control of the graphical representation of data is a fundamental requirement for any professional mapping community.

Styled layer Descriptor Implementation Specification, Open GIS Consortium (OGC)Inc., 2002-09-19, pp ix

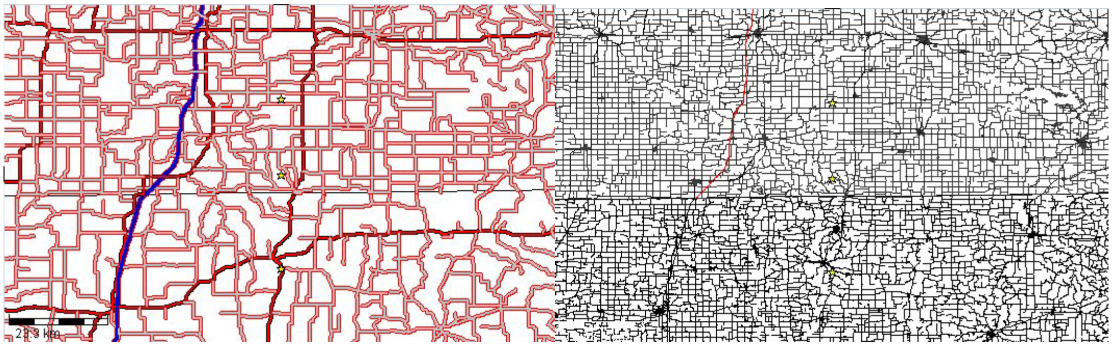
Introduction

A Styled Layer Descriptor (SLD) is an optional feature of an OGC-compliant mapserver. SLDs may be specified as part of the getmap request to compliant web map services. An SLD allows USGS-defined symbology to override the data provider's default symbology. In the case of *The National Map* Catalog, user-defined symbology is synonymous with USGS-defined symbology. By using SLDs to define the symbol(s), data sets from different data partners with different default symbologies can be displayed with consistent symbology.

An SLD is an XML document created at the time of the getmap request to override the default symbology placed on a data layer by the creator of the data. The integration of multiple data sets with various attribution schemes and symbologies is one of the challenges USGS must confront in order to maximize the usefulness of *The National Map*. SLDs are one means of accomplishing this integration.

For example, a series of road files covering a given geographic area, or adjoining geographic areas, may come from a variety of federal, state, and local sources. Each of these data sets probably will have a unique attribution scheme and symbology set. When these data sets are displayed simultaneously, the results can be confusing. SLDs create the potential to display all of the data sets with uniform symbols based on definitions defined by USGS. These definitions are made in the Catalog in the form of filters and symbolgroups. These require no additional work on the part of the data provider and do not change the default symbology the provider associated with the data; thus, the provider can continue to use its original symbology for non-*TNM* applications.

FIGURE 1:

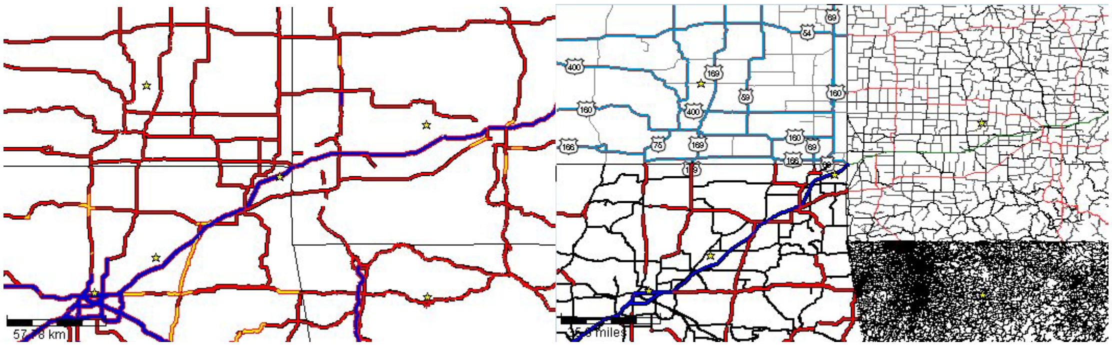


Missouri and Iowa roads with SLDs

Missouri and Iowa roads with default symbology

Although consistent symbology for differing data sets is possible, the use of SLDs does not guarantee that the data sets will match 100 percent. Different collection criteria, attribution styles, and temporal differences are a few of the possible reasons that complete integration may not be possible. However, SLDs will allow *TNM* users to see the extent to which the attribution models are complimentary.

FIGURE 2:



KS, MO, AR, and OK with SLD

Same data sets with default symbology

Symbology Manipulation

There are several ways to manipulate the description of the layers within a series of data sets so that they will display with consistent symbology:

1. Color all features the same way. This is a simple procedure but actually hides available information which may be beneficial to the user. (See AR area in Figure 2)
2. The default symbology in the data set itself can be changed to match that of a common USGS symbology set based on a defined standard. This is a labor intensive process that changes the original data. These changes could possibly lower the usefulness of the data for its intended use and require additional work on the part of the data provider. (See Oklahoma area in Figure 2, default symbology) This also requires updates any time there is a change to the symbology.
3. Styled Layer Descriptors (SLD) apply USGS-defined symbols to each data provider's data set. This allows for uniform display of similar features from different data providers. (See KS, MO, AR, and OK with SLD in Figure with SLD in figure 2) SLDs do not change the data-provider's original symbology.

Styled Layer Descriptors

Styled Layer Descriptors are created through XML encoding. This is referred to as StyledLayerDescriptor XML. The StyledLayerDescriptor XML allows for user (USGS) defined symbology and automatic legend generation. This is accomplished through the use of the SLDGenerator service, created at MCMC, which extracts and applies layer and style parameters specified in the Catalog.

The National Map Catalog is a database. Within this database are numerous tables that help define what data are available to *The National Map.* There are also a number of tables that exist to define SLDs. The SLDGenerator uses fields from both sets of tables within the Catalog database to define when the SLD will be implemented and what the symbology will look like. Table 1 lists several of the tables and fields in the Catalog database that relate to SLDs.

TABLE 1: Catalog Table and Field Examples

Table Name/Field Name	Purpose
elements.sldurl	Defines the location of a static SLD
elements.symbolgroupid	Joins Catalog schema to SLD schema
symbol_group.id	Specifies the symbol to be used
strokes.width	Defines the width of a symbol in pixels

There are two types of SLDs, either of which can be implemented through the Catalog. A "static" SLD is an XML document that can reside on any server. A "dynamic" SLD creates the XML document from information contained in the Catalog. If the information in the Catalog is changed then the XML document changes. The elements.sldurl field in the Catalog can direct the viewer to either static or dynamic SLDs. If the elements.symbolgroupid is also populated, the Catalog will direct the viewer to a dynamic SLD created by the SLDGenerator. The viewer adds the SLD to the XML getmap request. In either case, the viewer makes a getmap request to the service providing the data, and if the service has been SLD enabled[1], the SLD is implemented. The SLD parameters replace the default layer and style parameters specified in the original layer. The elements.sldurl field is also used when "layer parts" are involved.

An SLD can key on a single USGS-defined symbol_group.id if everything in the layer requires the same symbology. If no filter is defined, the interpretation is that this rule condition is always true and all features in the layer are accepted and styled the same way. A filter can key on a single or multiple attributes present in the dataset. The use of the Boolean operators (and, or, and not) in the filter specifies how the attributes will be interpreted.

SLDs may extract specific layer-parts from a layer. Layer-parts can be thought of as children extracted from the parent layer through the use of a specific parameter or a group of parameters, like attributes. This process involves the use of layer filters based on fields in the attribute table. Layer-parts can be tied to and extracted by scale dependencies again based on specific attributes. Layer-parts can also be extracted by feature type: point, line, or polygon. Table 2 shows of how a data layer may be broken into layer-parts. SYSCODE is simply an attribute field and the numbers are specific attributes.

[1] Minnesota MapServer services are OGC-Compliant and therefore SLD enabled. ArcIMS services require a connector in order to be SLD enabled. There are problems with the ArcIMS connectors supplied by ESRI. The most successful connector in use at this time is MCMC's OGC connector.

TABLE 2: Layers and Layer Parts

Parent Layer	Layer-Parts (Children)	Unique Identifier: Field/Attribute
Roads - Iowa	Iowa Interstates	SYSCODE=1
	Iowa US/Primary State Highways	SYSCODE=2
	Iowa Secondary State/County Roads	SYSCODE=3 OR SYSCODE=4
	Iowa Local Roads	SYSCODE=5
	Road Labels Iowa	NINEONEONE

The filtering process can be complicated by the lack of attribution associated with a given data set. It is possible that a data set will not have any attribute field that will serve adequately as the unique identifier necessary to create a layer part. Problems also can arise when different data producers use attribution schemes that are not compatible.

It is not necessary to have a single common attribute to extract a layer-part, but something unique must identify the features to be symbolized. Multiple attributes can be used along with Boolean operators. For example, given an attribute field of "type" for a road layer containing attributes "class 1", "class 2", and "class 3", a filter could be created to extract just the "class 2" roads:

type = class 2

Or, a filter could be created to extract the class 1 and class 2 roads

type = class 1 OR type = class 2

Or, a filter could be created to exclude "class 1" roads:

type not "class 1"

Static Styled Layer Descriptors

There are two basic types of Styled Layer Descriptors, static and dynamic. Both types are used to over ride the default symbology from the data provider.

A static SLDURL must be specified for each data layer in the Catalog whose symbology is to be overridden. The URL may reside on USGS, data partner, or other Web servers.

Static SLDs have several disadvantages:

1. Symbol definitions are not normalized in a single place requiring symbol changes to be made in multiple locations.
2. Scale-dependencies and layer-part filters are hard-coded in each SLD document and not easily maintained or available to other applications.
3. Legend graphics must be refreshed every time the SLD changes.[2]

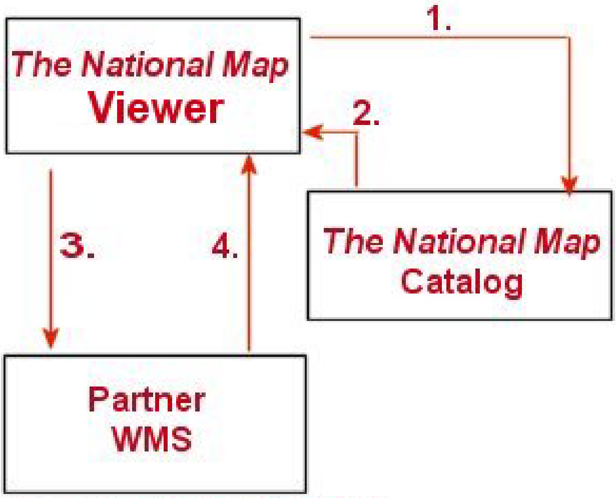
[2] Consistent Symbology with Styled Layer Descriptors, Larry Moore, PowerPoint Presentation, 2004

Dynamic Styled Layer Descriptors

Because of the disadvantages of static SLDs, the Catalog has been designed to support dynamic SLD and legend generation. Symbols are defined in tables within the Catalog's SLD schema and a single "symbol_groups.id" is applied to each layer or "layer part" to be symbolized. This ID ties Catalog defined symbols to the data layers.

The viewer issues a getmap request to the service. The request includes a SLDGenerator created URL which points to a USGS-defined symbol set, or SLD. The SLD is defined in tables in the Catalog. An SLD XML file is generated using these definitions from tables in the Catalog. The service uses the SLD to override the default symbology and replace it with the USGS-defined symbology.

FIGURE 3



1. application queries the Catalog
2. Catalog returns map info to application, with SLD
3. application sends getmap request to service
4. map with SLD symbology returned to application

Dynamic SLDs offer the following advantages over static SLDs

1. Symbol definitions are normalized in the Catalog simplifying maintenance and sharing issues. A symbol change in the catalog can instantly change the symbology for any number of related layers.
2. Scale-dependency can be set within the Catalog allowing symbology to change with scale.
3. Layer-part filters, which can be used to symbolize different features within a layer based on attributes, are defined within the Catalog.

4. Layer-part control of presentation allows for the display or non-display of features within a layer based on criteria set in the Catalog. Part of a layer can be displayed while other parts are turned off.
5. Legends can be dynamically created from the symbols created by the SLDs through the Catalog. Changes to the symbology defined in the Catalog automatically result in changes to legends.[3]

Basic model of a Catalog Defined SLD

The stroke is the basic element of a symbol. The stroke is defined by the color, opacity, and width of the symbol. A line symbol can be made from a single stroke. In turn, a single stroke, or any number of strokes can be combined into a symbol group that dictates what will be displayed.

- A specific layer has one or more symbol_groups
  A layer can be broken into multiple user-layers, each of which has one and only one symbol _group.
- A symbol_group is represented by one symbol_group_entries.
- A symbol_group_entries contains one or more symbols
- A symbol may be a part of zero or many symbol_group_entries.
  Symbols for points, lines, and polygons are stored in separate tables.
- A symbol has one stroke.
- A stroke has one color, opacity, and width,
  Polygon and point symbols will also have a fill. Point symbols also have a graphic_part.

Table 3: Components of Symbols

Symbol type	Components
Line	Stroke
Polygon	Stroke, fill
Point	Fill, graphic, size
Text	Label, font, label placement, halo, fill

For example, a road can have a symbol created from a single stroke. A line_symbol could be created with a stroke that was red, 100 percent opaque, and two pixels wide. The line_ symbol is then used to populate the symbol_group in the symbol_group_entries table. The symbol_group in turn determines what will be visible to the user. In another instance, a road can have a symbol created from multiple strokes. Another line_symbol could be created that was black, 100 percent opaque, and four pixels wide. The two then can be joined and ordered by the symbol_group_entries table. The result is a two pixel wide red road with a one pixel wide casing on each side.[4]

[3] Originally from SLDIntro, Larry Moore, PowerPoint, 2004, several changes and additions.
[4] See Appendix A: an SLDXML document showing symbol definitions.

The Z-order of multiple symbols is specified in the symbol_group_ entries table. The lowest-numbered symbol on the list is plotted first and higher-numbered symbols are plotted in succession.

Current SLD Related Problems

Connectors

In order for SLDs to work, the service must be SLD-enabled. OGC-compliant map servicers are enabled, and Mapserver is OGC compliant. ArcIMS achieves OGC compliance through a connector. The connector allows the symbology defined by an SLD to over-ride the default symbology established by the data producer for a particular getmap request. At this time, many data providers are using ArcIMS and have not enabled their SLD connectors. There are also problems with the ArcIMS connectors supplied by ESRI. Presently, the only dependable connector appears to be the OGCConnector created by USGS' Mid-Continent Mapping Center.

Symbology

Certain symbols do not work well with certain flavors of Web Map Services. Dashes and dots are prime examples of this problem. A composite symbol that includes dashes, like a railroad, is easy to represent in SLD, but map servers are not consistent in how they handle dashed symbols. This may result in inadequate and inconsistent symbols on the display. The concept of dashed and dotted symbology may need to be reviewed. Symbols that work well on a fixed-scale graphic map may not be suitable for multiple-scale digital maps.

It also should be remembered that traditional map symbols were created for fixed-scale maps. Scales are not fixed in a digital environment, and, therefore, traditional dashed and dotted symbols may be difficult to read and thus may not be suitable. Solid-line symbology appears to be better suited for a digital mapping environment.

Attribution

Data-producers attribute data for their own needs. Those needs do not always correlate with the needs of another data-producer supplying similar data over the same or an adjoining area. Because of this it may not always be possible come up with symbology sets that match all attribute schemes 100 percent. Due to differences in naming conventions it also takes some investigation to determine which attribute fields to use when creating filters.

Data sets have also been found that do not have an adequate unique identifier or set of identifiers. In this case it is not possible to symbolize the data without a significant amount of extra work.

WMS Differences

There are a variety of Web Map Services, and they do not all act exactly alike. ArcIMS and MapServer, for instance do not handle scales the same way. AcrIMS represents scales in the same way as the Catalog (pixel/degrees), but MapServer uses a scale denominator. This causes the viewer to display the toggle for a layer at times when the layer is not available. Other differences also have been found.

Conclusion

Styled Layer Descriptors have the potential to provide a relatively consistent graphical representation of digital map data served from a wide variety of data-providers. The cooperation of the partner providing the data is essential because their service must be SLD enabled and the data must have a means of uniquely identifying feature types. The USGS data theme coordination teams must develop a set of symbols and rules regarding the display of data in *The National Map* Viewer. The implementation of SLD can be accomplished with this cooperation. Without such cooperation, however, implementation of SLDs will be difficult.

Appendix A

The following is a SLD XML created by SLDGenerator. Notes in **BOLD** capital letters and underlined have been added for clarification and are not a part of the XML document. The first part describes a three pixel blue line with one pixel casings.

```
  <?xml version="1.0" encoding="UTF-8" ?>
- <StyledLayerDescriptor 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"
      version="1.0.0">
  - <NamedLayer>
      <Name>NTAD_roads</Name>     LAYER NAME
    - <UserStyle>
      - <FeatureTypeStyle>
        - <Rule>            SYMBOL DEFINITION  BEGINS HERE
          - <ogc:Filter>
            - <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>SIGNT1</ogc:PropertyName>  ATTR. FIELD
                <ogc:Literal>I</ogc:Literal>  ATTRIBUTE
              </ogc:PropertyIsEqualTo>
            </ogc:Filter>
            <MaxScaleDenominator>16772987</MaxScaleDenominator>  MAX VIEWSCALE
          - <LineSymbolizer>
            - <Geometry>
                <ogc:PropertyName>Default</ogc:PropertyName>
              </Geometry>
            - <Stroke> FIRST PART OF SYMBOL – 5 PIXEL RED LINE
                <CssParameter name="stroke">#FF0000</CssParameter>  COLOR
                <CssParameter name="stroke-opacity">1.0</CssParameter>
                <CssParameter name="stroke-width">5.0</CssParameter>  PIXELS
                <CssParameter name="stroke-linejoin">mitre</CssParameter>
                <CssParameter name="stroke-linecap">butt</CssParameter>
              </Stroke>
            </LineSymbolizer>
          - <LineSymbolizer>
            - <Geometry>
                <ogc:PropertyName>Default</ogc:PropertyName>
              </Geometry>
            - <Stroke> SECOND PART OF SYMBOL -  3 PIXEL BLUE LINE
                <CssParameter name="stroke">#0000C8</CssParameter>
                <CssParameter name="stroke-opacity">1.0</CssParameter>
                <CssParameter name="stroke-width">3.0</CssParameter>
                <CssParameter name="stroke-linejoin">mitre</CssParameter>
                <CssParameter name="stroke-linecap">butt</CssParameter>
              </Stroke>
            </LineSymbolizer>
          </Rule>           SYMBOL DEFINITION ENDS HERE
        - <Rule>            SECOND SYMBOL DEFINITION BEGINS HERE
          - <ogc:Filter>
            - <ogc:Or>
              - <ogc:Or>
                - <ogc:Or>
                  - <ogc:PropertyIsEqualTo>
                        <ogc:PropertyName>SIGNT1</ogc:PropertyNam
                        e>
                      <ogc:Literal>S</ogc:Literal>
                    </ogc:PropertyIsEqualTo>
                  - <ogc:PropertyIsEqualTo>
                        <ogc:PropertyName>SIGNT1</ogc:PropertyNam
                        e>
                      <ogc:Literal>C</ogc:Literal>
                    </ogc:PropertyIsEqualTo>
                  </ogc:Or>
                - <ogc:PropertyIsEqualTo>
                    <ogc:PropertyName>SIGNT1</ogc:PropertyName>
                    <ogc:Literal>F</ogc:Literal>
                  </ogc:PropertyIsEqualTo>
                </ogc:Or>
              - <ogc:PropertyIsEqualTo>
                  <ogc:PropertyName>SIGNT1</ogc:PropertyName>
                  <ogc:Literal>H</ogc:Literal>
                </ogc:PropertyIsEqualTo>
              </ogc:Or>
            </ogc:Filter>
            <MaxScaleDenominator>2515948</MaxScaleDenominator>
          - <LineSymbolizer>
            - <Geometry>
                <ogc:PropertyName>Default</ogc:PropertyName>
              </Geometry>
            - <Stroke>
                <CssParameter name="stroke">#FF0000</CssParameter>
                <CssParameter name="stroke-opacity">1.0</CssParameter>
                <CssParameter name="stroke-width">4.0</CssParameter>
                <CssParameter name="stroke-linejoin">mitre</CssParameter>
                <CssParameter name="stroke-linecap">butt</CssParameter>
              </Stroke>
            </LineSymbolizer>
          - <LineSymbolizer>
            - <Geometry>
                <ogc:PropertyName>Default</ogc:PropertyName>
              </Geometry>
            - <Stroke>
                <CssParameter name="stroke">#C0C0C0</CssParameter>
                <CssParameter name="stroke-opacity">1.0</CssParameter>
                <CssParameter name="stroke-width">2.0</CssParameter>
                <CssParameter name="stroke-linejoin">mitre</CssParameter>
                <CssParameter name="stroke-linecap">butt</CssParameter>
              </Stroke>
            </LineSymbolizer>
          </Rule>
        - <Rule>
          - <ogc:Filter>
            - <ogc:PropertyIsEqualTo>
                <ogc:PropertyName>SIGNT1</ogc:PropertyName>
                <ogc:Literal>U</ogc:Literal>
              </ogc:PropertyIsEqualTo>
            </ogc:Filter>
            <MaxScaleDenominator>8386493</MaxScaleDenominator>
          - <LineSymbolizer>
            - <Geometry>
                <ogc:PropertyName>Default</ogc:PropertyName>
              </Geometry>
            - <Stroke>
                <CssParameter name="stroke">#000000</CssParameter>
                <CssParameter name="stroke-opacity">1.0</CssParameter>
                <CssParameter name="stroke-width">4.0</CssParameter>
                <CssParameter name="stroke-linejoin">mitre</CssParameter>
                <CssParameter name="stroke-linecap">butt</CssParameter>
              </Stroke>
            </LineSymbolizer>
          - <LineSymbolizer>
            - <Geometry>
                <ogc:PropertyName>Default</ogc:PropertyName>
              </Geometry>
            - <Stroke>
                <CssParameter name="stroke">#FF0000</CssParameter>
                <CssParameter name="stroke-opacity">1.0</CssParameter>
                <CssParameter name="stroke-width">2.0</CssParameter>
                <CssParameter name="stroke-linejoin">mitre</CssParameter>
                <CssParameter name="stroke-linecap">butt</CssParameter>
              </Stroke>
            </LineSymbolizer>
          </Rule>
        </FeatureTypeStyle>
      </UserStyle>
    </NamedLayer>
  </StyledLayerDescriptor>
```