Towards a Declarative Portrayal and Interaction Model for GIS and LBS

Thomas Brinkhoff
Institute for Applied Photogrammetry and Geoinformatics (IAPG)
FH Oldenburg/Ostfriesland/Wilhelmshaven (University of Applied Sciences)
Ofener Straße 16/19, D-26121 Oldenburg, Germany
Thomas.Brinkhoff@fh-oldenburg.de

SUMMARY

An essential property of GIS is the adequate visual portrayal of spatial data. The OGC Styled Layer Descriptor (SLD) specification is a map-styling language for producing geo-referenced maps with user-defined styling. However, SLD shows some weaknesses: 1.) it does not support interaction, 2.) it does not consider the requirements of mobile GIS, and 3.) it is rather restricted in referencing objects. Therefore, we propose a portrayal and interaction model that can be applied to traditional GIS as well as to mobile applications. SLD is used as the base specification especially extended by characteristics of the W3C Cascading Style Sheets specification and by observing the demands of LBS. Another aspect that has influenced the design of our model is its applicability to XML applications like Scalable Vector Graphics (SVG).

KEYWORDS: *Portrayal Model, Styling, Interaction, LBS, Mobile GIS*

INTRODUCTION

An essential property of GIS is the adequate visual portrayal of spatial data. *Portrayal properties* define, for example, colors, the size of symbols and fill styles. Typically, it is possible to describe such properties for particular layers and scale ranges. *Interaction properties* determine the behavior of objects with respect to selection, modification etc.

Location-based Services (LBS) use the current position of the user to determine the information that should be requested and presented. If GIS operations are offered, the corresponding system is called *Mobile GIS*. Mobile GIS and LBS have special demands on the presentation of maps and on the interaction with spatial objects. These additional requirements result from the varying position and orientation of the user and from the typical applications performed on mobile devices.

The definitions of the portrayal and interaction properties by commercial GISs are currently not compatible. Regarding current OGC specifications, the *Styled Layer Descriptor (SLD) Implementation Specification* (OGC, 2002a) is the relevant standard. It defines a formal setting that allows defining styling properties. In SLD, *symbols* represent the visualization of a feature's geometry. However, SLD has some weaknesses: 1.) it does not support interaction, 2.) it does not consider the requirements of mobile GIS, and 3.) it is rather restricted in referencing objects. The *Extensible Stylesheet Language* (W3C, 2001) and *Cascading Style Sheets* (W3C, 1998) are other relevant approaches.

In the following, we propose a *portrayal and interaction model* that can be applied to traditional GIS as well as to mobile GIS. This model is based on SLD, but in addition we

- enhance the set of referable objects,
- introduce new properties with respect to interaction,
- add so-called external parameters (like the current location and speed) that influence the portrayal and interaction properties, and
- support rules that determine the styling and interaction dependent on external parameters.

External parameters are especially helpful for mobile GIS. Another aspect, which has influenced the design of our model, is its applicability to XML applications like the SVG (W3C, 2003) and to mobile geospatial SVG viewers (Brinkhoff, 2003).

In this work, we assume that a map consists of *features* each defined by a *feature type*. The location of a feature is specified by one spatial attribute. Further spatial properties may be stored in additional attributes. A feature type may establish additional non-spatial attributes. A feature belongs to one *layer* storing features of the same feature type.

RELATED WORK

There exists an extensive work about the visualization of geospatial data. For overviews, see e.g. Kraak & Ormeling (2003) and Hake et al. (2002). However, our goal is not to propose new cartographic techniques or certain representations for real-world phenomena. Instead, our portrayal and interaction model should enable an application to attach style and interaction properties to features in a manner that is declarative, that conforms to current data representations (like XML) and that observes current standards. Therefore, the following discussion of related work is focused on presenting and assessing standards that have relevance as IT technologies.

Cascading Style Sheets

For web applications, *Cascading Style Sheets (CSS)* allow authors to attach style to structured documents (e.g., HTML documents and XML applications). By separating the presentation style of documents from the content of documents, CSS simplifies web authoring and site maintenance. The current version of CSS is level 2 *(CSS2)* (W3C, 1998).

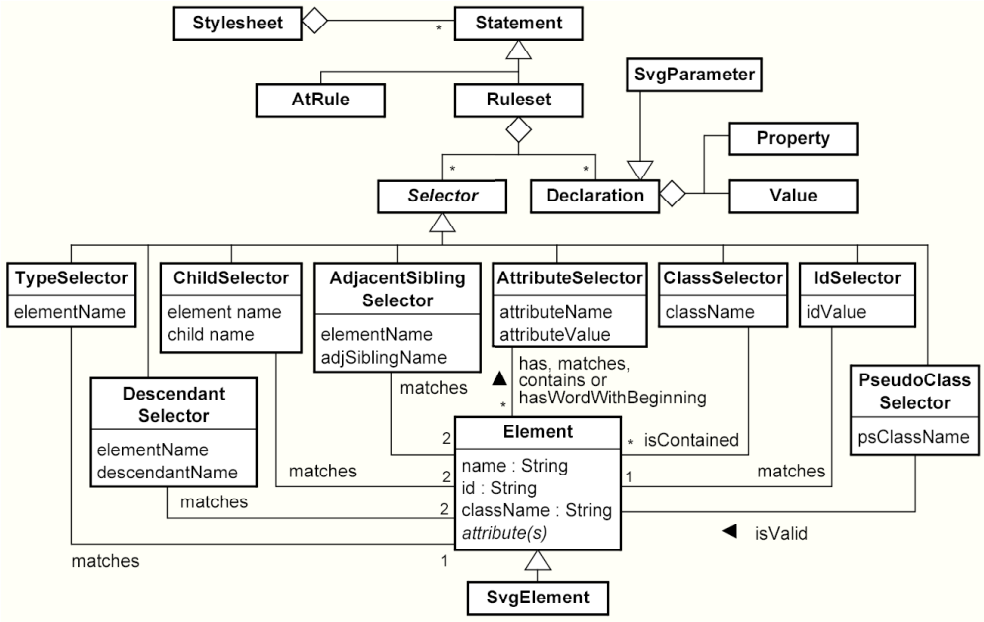

Figure 1: Simplified data model of CSS2.

Figure 1 depicts a simplified UML class diagram of CSS2. A stylesheet contains a list of *statements*. Statements are mostly so-called *rulesets*. A ruleset consists of *selectors* and *declarations*. In Figure 2, `text` and `rect.house`, e.g., are selectors and the parts delimited by curly brackets after the selectors contain the declarations. Each declaration consists of the name of a *property* and its value. Selectors allow determining elements for that a declaration should be applied. *Type selectors* match the name of

elements (e.g., `rect` and `text` in 0). *Attribute selectors* allow selecting elements that have the particular attribute or that have the particular attribute matching or containing or starting with a defined attribute value (like `line[x1="10"]` in Figure 2). A special case of attribute selectors are *class selectors* and *ID selectors*. Class selectors determine elements with matching `class` attributes; they can be applied to any element (`.important`) or only to special elements (`rect.house`). ID selectors look for an element having the specified `id` attribute. The descendant selector, the child selector and the adjacent sibling selector support the tree structure of the Document Object Model (DOM) that represents an XML or an HTML document.

```
@media all {
  rect { fill: red; stroke: blue; stroke-width: 1px; }
  rect.house { fill: yellow; stroke: rgb(255,200,0); }
  .important { fill: none; stroke: green; }
  text { font-family: sans-serif; stroke: red; opacity: 1; }
  line[x1="10"] { fill: none; stroke: black; }
  :hover { stroke: red; }
}
@media printer {
  rect { stroke-width: 5px; }
}
```

Figure 2: Example of a CSS stylesheet.

For interaction purposes, the *pseudo-class selector* is of special interest. The `:hover` pseudo class applies while the user designates an element (with some pointing device), but does not activate it. The `:active` pseudo class is valid while an element is being activated by the user (e.g. by pressing a mouse button). The `:focus` pseudo class applies while an element has the focus.

At-rules allow defining CSS properties that are only designed for certain media. Examples for *media types* listed in (W3C, 1998) are handheld devices, printers and computer screens. In 0, the `@media all` includes the styles for all media types, whereas `@media printer` defines special properties holding for printers.

Extensible Stylesheet Language (XSL)

The *Extensible Stylesheet Language (XSL)* (W3C, 2001) is a language for expressing stylesheets. Given a class of arbitrarily structured XML documents, designers use an XSL stylesheet to express their intentions about how that structured content should be presented; that is, how the source content should be styled, laid out, and paginated onto some presentation medium. The formatting process comprises several steps, some of which depend on others in a non-sequential way. XSL formatting is suitable for XML applications. As a declarative programming language, XSL gives developers a lot of flexibility in coding but does not support the definition of time-efficient evaluation strategies. Therefore, the processing of XSL is often rather costly for large XML documents (Bittner 2004). Furthermore, XSL does not support graphical shapes and does not consider interaction aspects. Because of these restrictions we do not consider XSL in the rest of this paper.

Styled Layer Descriptors

The OGC has proposed the *Styled Layer Descriptor (SLD) Implementation Specification* (OGC, 2002a). This document specifies the format of a map-styling language for producing geo-referenced maps with user-defined styling. A slightly modified and extended version of SLD has been proposed by the OGC Discussion Paper *Style Management Service (SMS)* (OGC, 2003a). Figure 3 shows an extract of SLD according to this version (OGC, 2003a) using UML.

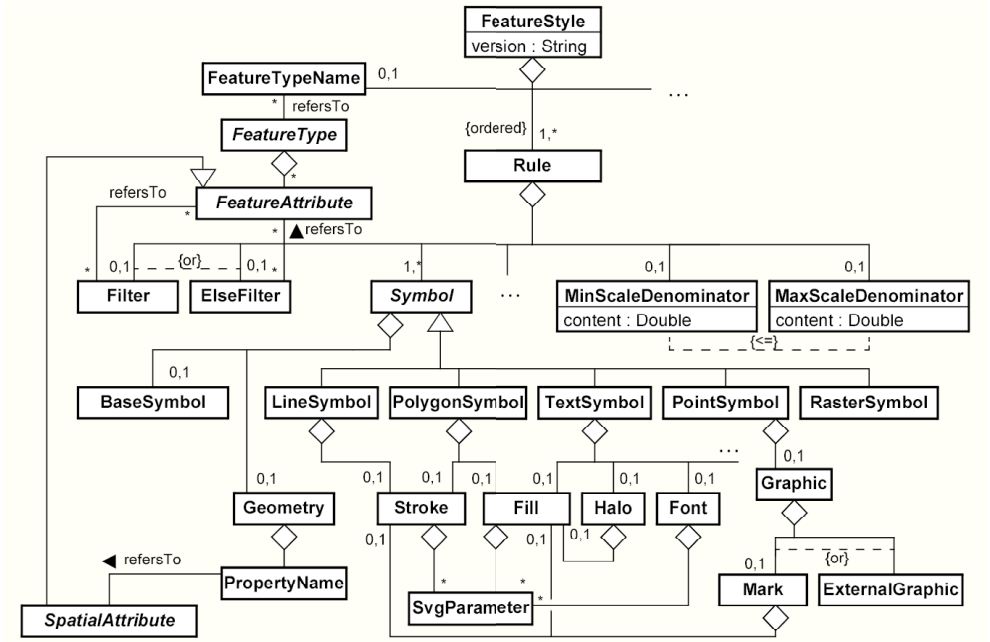

Figure 3: Simplified Data Model of SLD according to (OGC 2003a).

The style definitions of SLD are connected to feature types: `FeatureStyle` elements reference the name of feature types. *Rules* are used to select rendering instructions. A `FeatureStyle` may have several rules. In this case, they are ordered according to their "priority". Conditions concerning feature properties can be defined by a `Filter` element (OGC, 2001). The *OGC Filter Encoding Specification* allows using the geometric and topological operators of the *OGC Simple Features Specification* (OGC, 1999). Instead of the `Filter` element, a rule can use the `ElseFilter` element. SLD also supports conditions concerning the visualization: the `MinScaleDenominator` and `MaxScaleDenominator` define the scale range for a rule. A rule holds, if both, the filter and the scale range hold. Therefore, the same filter element and / or scale range may occur in more than one rule of the same `FeatureStyle`.

Symbols are embedded inside of rules. They describe how a feature is to appear on a map. Five types of symbols exist: `LineSymbol`, `PolygonSymbol`, `PointSymbol`, `TextSymbol`, and `RasterSymbol`. Their properties can be described by one or more CSS parameters defined by the SVG specification (W3C, 2003). On this level, a further selection may be performed: the `Geometry` element allows the specification of the name of the feature attribute (`PropertyName`) storing the affected geometry (OGC, 2002b).

Figure 4 shows an example using the (OGC, 2003a) specification.

```
<FeatureStyle version="1.0.20" xmlns="http://www.opengis.net/sld"
              xmlns:ogc="http://www.opengis.net/ogc">
  <FeatureTypeName>River</FeatureTypeName>
  <Rule>
    <Name>SmallRivers</Name>
    <ogc:Filter>
      <ogc:PropertyIsEqualTo>
        <ogc:PropertyName>RiverClass</ogc:PropertyName>
        <ogc:Literal>1</ogc:Literal>
      </ogc:PropertyIsEqualTo>
    </ogc:Filter>
    <LineSymbol>
      <Geometry> <ogc:PropertyName>center-line</ogc:PropertyName> </Geometry>
      <Stroke>
        <SvgParameter name="stroke">#aaaaff</SvgParameter>
        <SvgParameter name="stroke-width">5.0</SvgParameter>
      </Stroke>
    </LineSymbol>
  </Rule>
  <Rule>
    <Name>LargeRivers</Name>
    <ElseFilter />
    <LineSymbol>
      <Geometry> <ogc:PropertyName>center-line</ogc:PropertyName> </Geometry>
      <Stroke>
        <SvgParameter name="stroke">blue</SvgParameter>
        <SvgParameter name="stroke-width">10.0</SvgParameter>
      </Stroke>
    </LineSymbol>
  </Rule>
</FeatureStyle>
```

Figure 4: SLD example.

ASSESSMENT

The previous examination allows the conclusion that existing models are not sufficient to describe the complete behavior of current GIS. Furthermore, mobile GIS are completely ignored.

	SLD	CSS2
referencing		
of layers	+ (by FeatureTypeName)	+ (by TypeSelector)
of spatial attributes	+ (by PropertyName)	+ (by TypeSelector and others)
of parts of DOM	-	+
filtering		
by non-spatial attributes	+ (by Filter and ElseFilter)	± (by AttributeSelector)
by spatial attributes	+ (Filter Encoding Spec.)	-
by LBS conditions	-	-
by visualization conditions	± (only current scale)	± (only media types)
definition		
of portrayal properties by SVG	+	+
of interaction properties	-	± (by PseudoClassSelector)

Figure 5: Comparison of SLD and CSS.

The table in Figure 5 compares the most important properties of SLD and CSS2. Both approaches allow referring to layers and spatial attributes. However, *referencing* of data in specific parts of the hierarchical DOM is only supported by CSS2; SLD does not support query conditions that refer to the structure or hierarchy of documents. Regarding the *filtering* of features according to their attributes, SLD has essential advantages: its `Filter` element is more powerful than the CSS2 `AttributeSelector` and supports geometric and topological filter conditions whereas CSS2 supports only simple alphanumerical comparisons. The support of conditions referring to the state of visualization is poor in both approaches because the state of the user agent is not considered.

Exceptions are the current scale (SLD) and media types (CSS2). Both, CSS2 and SLD allow *defining* style properties by SVG attributes. Interaction is not (SLD) or is only poorly supported by CSS2 pseudo-class selectors.

REFERENCING SUBJECTS OF RULES

According to the previous evaluation, the most promising approach is to use the concepts of SLD as base for the development of a declarative model that describes the portrayal and interaction properties of GIS and mobile applications. This starting point will be extended by CSS and LBS characteristics.

In our model, rules can generally be defined (a) for an individual feature, (b) for all features of a class, (c) for all features of a feature group, or (d) for all features of the same feature type (layer). A *feature group* is an aggregation of features. In contrast to a layer, the features of one group may have different feature types. A feature belongs at most to one group directly, but a group may be grouped with other groups or features. Feature groups allow modeling object hierarchies like the group element in SVG or other XML applications. A *class of features* consists of all features with `class` attributes of the same value; the features of a class may have all the same feature type or have different feature types.

Figure 6 depicts the relevant part of the SLD model: the specification of the `FeatureStyle` element. It is augmented by elements allowing additional references. In contrast to the SLD specification, the elements `Class` and `ID` have been added. The element `Class` addresses all features and / or feature groups that have a `class` attribute of a certain value. The `ID` element selects one feature or one feature group. The new elements are related to the CSS `ClassSelector` and `IdSelector`. Further additions are the `PseudoClass` and the `MediaType` element. They have the same purpose as the corresponding CSS elements. Besides of traditional media types, types like "main map" and "overview map" are possible defining different styles and visibilities for the main map window and for an overview map. Figure 7 shows an example.

Using our model, a feature can be referenced as subject by more than one `FeatureStyle`. Then, the styling for a particular feature is combined from feature styles having different types of subjects. In case of contradictions, the properties for (a) have the highest rank and properties for (d) have the lowest significance.

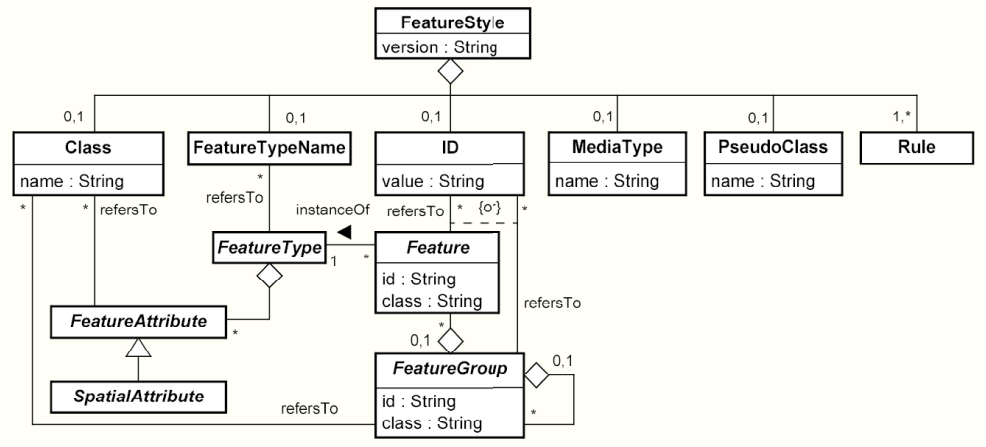

Figure 6: Supplemented `FeatureStyle` element.

```
<FeatureStyle version="...">
  <Description><Abstract>Styling for River features of class "c1"</Abstract></Description>
  <FeatureTypeName>River</FeatureTypeName>
  <Class>c1</Class>
  <Rule> ... </Rule>
</FeatureStyle>
<FeatureStyle version="...">
  <Description><Abstract>Styling for feature with ID "_4303000"</Abstract></Description>
  <ID>_4303000</ID>
  <Rule> ... </Rule>
</FeatureStyle>
<FeatureStyle version="...">
  <Description><Abstract>Styling for selected symbols on main map</Abstract></Description>
  <MediaType>main_map</MediaType>
  <PseudoClass>:highlight</PseudoClass>
  <Rule> ... </Rule>
</FeatureStyle>
```

Figure 7: Examples for new references.

EXTERNAL PARAMETERS

The attributes of features may influence the portrayal and interaction properties of its symbols. In addition, these properties may be affected by parameters that are not attributes of the corresponding feature or by any feature at all. These parameters are called *external parameters* in the following.

Analogous to the `Filter` and `ElseFilter` elements, a rule is extended by the new `EPFilter` and `EPElseFilter` elements. They refer to arbitrary external parameters. The syntax of `EPFilter` is the same as the syntax of Filter, except that the names of external parameters replace the names of feature properties. These two new elements replace the more restricted SLD elements `MinScaleDenominator` and `MaxScaleDenominator`. Like those two elements, one `EPFilter` or `EPElseFilter` can be used in a rule in addition to a `Filter` or an `ElseFilter` element (see Figure 8). An example is depicted in Figure 9.

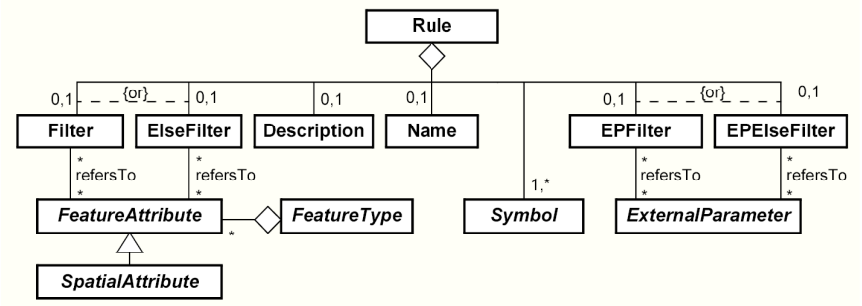

Figure 8: Usage of external parameters in rules.

General External Parameters for Traditional GIS

All GIS allow defining the visibility depending on the *current scale*. In order to support this behavior, the current scale is given by the external parameter `curr-scale`. It is a numerical value defined by the denominator following the SLD specification (OGC, 2002a). The viewport currently depicted by the GIS is another important external parameter. It is, for example, required for reloading parts of a map. The current viewport `curr-viewport` is given as a Simple Feature Geometry (OGC, 1999).

External Parameters for Advanced GIS

Time is a very important dimension for visualizing spatial data (Langran, 1993) (Kraak & Ormeling, 2003). In contrast to dynamic variables (Kraak & MacEachren, 1994), external parameters reflect

outer events. Thus, `curr-date` giving the *current date* and *current time* is a suitable external parameter for LBS. For mobile applications, for example, the visualization of symbols should depend on the time of day.

For LBS, the *current location* is of highest interest. The external parameter `curr-position` describes the position as a two-dimensional Simple Feature `Point`. In addition, the current altitude is given by the external parameter `curr-altitude`. Considering the ideas of Max Egenhofer (2004) about mobile space, also the *current speed* of an LBS user should influence the map visualization. For example, the selection and amount of data that a car driver can perceive depends on the speed of the car. For supporting such dependencies, the external parameter `curr-speed` is introduced.

Another important (future) property of mobile devices is its orientation. It will be reasonable to adapt the orientation of the map display to the *current orientation* of the device (Pammer & Radoczky, 2002). The external parameter `curr-orientation` gives this information as a numerical value.

Figure 9 shows two rules that use external parameters.

```
<FeatureStyle version="...">
  ...
  <Rule>
    <Description><Abstract>Styling dependent on current scale</Abstract></Description>
    <EPFilter>
      <PropertyIsGreaterThan>
        <EPName>curr-scale</EPName> <Literal>100000<Literal>
      <PropertyIsGreaterThan>
    </EPFilter>
    <LineSymbol>...</LineSymbol>
  </Rule>
  <Rule>
    <Description><Abstract>Styling dependent on current position</Abstract></Description>
    <EPFilter>
      <DWithin>
        <PropertyName>geom</ogc:PropertyName>
        <EPName>curr-position</EPName> <Distance unit="#meters">10</Distance>
      </DWithin>
    </EPFilter>
    <LineSymbol>...</LineSymbol>
  </Rule>
</FeatureStyle>
```

Figure 9: Example using external parameters.

STYLE AND INTERACTION PROPERTIES

Style Properties of SVG

SLD allows defining the properties of the `Stroke` and the `Fill` element by SVG parameters. Those parameters define the graphical rendering of symbols by border colors, fill patterns, line widths etc. More thrilling SVG parameters refer to the visibility and existence of symbols.

`visibility` is a style attribute of SVG. It allows excluding spatial objects from being visualized by the user agent. It has two values with an explicit meaning: `visible` means that the current symbol is visible and `hidden` that it is invisible. The `visibility` property refers to symbols that exist for the user agent. In extension to visibility, it can be distinguished between existent and non-existent symbols. This is supported by the SVG attribute `display`. It has two values with a relevant meaning: `inline` means that the current symbol exists for the user agent and `hidden` that it does not exist for the user agent; such elements cannot receive any events and are not considered by calculations.

```
<FeatureStyle version="...">
  ...
  <Rule>
    <Description><Abstract>Inside preload area</Abstract></Description>
    <EPFilter>
      <DWithin>
        <PropertyName>raster-area</ogc:PropertyName>
        <EPName>curr-position</EPName> <Distance unit="#meters">2000</Distance>
      </DWithin>
    </EPFilter>
    <RasterSymbol>
      <Style> <SvgParameter name="display">inline</SvgParameter> </Style>
    </RasterSymbol>
  </Rule>
  <Rule>
    <Description><Abstract>Outside preload area</Abstract></Description>
    <EPElse />
    <RasterSymbol>
      <Style> <SvgParameter name="display">hidden</SvgParameter> </Style>
    </RasterSymbol>
  </Rule>
</FeatureStyle>
```

Figure 10: Examples for scale- and position-dependent styling.

A user agent does not need to load and store symbols belonging to a non-existent group or layer. Changing `display` from `hidden` to `inline`, requires that the corresponding unloaded symbols are loaded. Changing it from `inline` to `hidden`, allows the user agent to unload the corresponding symbols. Non-existent symbols are especially important for mobile GIS getting data via wireless network connections and having only restricted memory capabilities. Figure 10 depicts a corresponding example; it is assumed that the area covered by raster maps is defined by a spatial attribute `raster-area`.

Non-SVG Style and Interaction Properties of Symbols

In addition to existing SVG attributes, other style and interaction properties are reasonable for GIS applications. They can easily be integrated into the SLD model by allowing additional attributes. Figure 11 indicates this by the inheritance of `SvgParameter`. A `Style` element has been added to `RasterSymbol` and `ExternalGraphic` in order to allow controlling visibility, existence and further attributes.

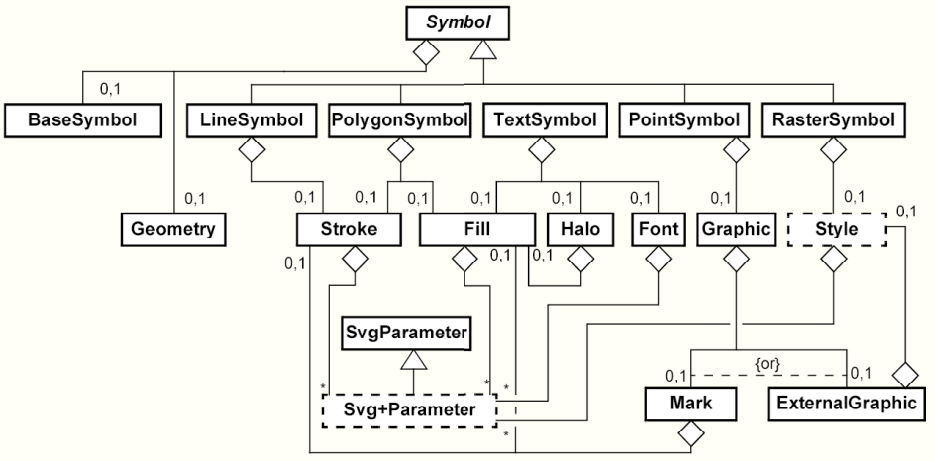

Figure 11: Integration of additional parameters by "SvgParameter" and "Style".

Lines are mostly drawn after filled polygons; polygons should typically be displayed in front of raster images. For defining and changing such behavior, a *display rank* is required. A display rank is currently not supported by SVG. Therefore, we introduce a new style attribute `rank`. It defines the order of drawing by a numerical value.

Other interaction properties may define the selectivity of features, their appearance in a legend, or their use by tool tips.

CONCLUSIONS

In this paper, a portrayal and interaction model has been proposed that can be applied to traditional GIS as well as to mobile applications. The model is declarative and considers the demands of XML applications. The OGC standard SLD was used as the base specification. It was extended by characteristics of the CSS2 specification and by concepts that allow observing the demands of LBS.

Future work consists mainly of two tasks: 1.) to transfer the model to an extended SVG specification and 2.) to expand an existing prototype of a mobile geospatial SVG viewer (Brinkhoff & Weitkämper, 2005). The first task consists of (a) the definition of rules in SVG documents and (b) the introduction of references to the rules into CSS2 style definitions. The second task should demonstrate that the proposed portrayal and interaction model can be implemented and allows an investigation of performance issues. Furthermore, the applicability of the model should be demonstrated by case studies.

BIBLIOGRAPHY

Bittner T.: Performance Evaluation for XSLT Processing, Techn. Report, University of Rostock, 2004.

Brinkhoff T.: A Portable SVG-Viewer on Mobile Devices for Supporting Geographic Applications, in: Proceedings 6th AGILE Conference on Geographic Information Science, Lyon, France, pp. 87-96, 2003.

Brinkhoff T., Weitkämper J., 2005: Mobile Viewers based on SVG$^{\pm geo}$ and XFormsGI, AGILE 2005.

Egenhofer M., 2004: Going Mobile with GIS, Invited Talk, 3. GI-Tage, Münster, Germany, 2004, http://www.spatial.maine.edu/~max/talks.html

Hake G., Grünreich D., Meng L.: Kartographie, 8. Aufl., Walter de Gruyter, 2002.

Kraak M.-J., MacEachren A.M.: Visualization of the Temporal Component of Spatial Data, in: Proceedings 6th International Symposium on Spatial Data Handling, Edinburgh, UK, pp. 391-409, 1994.

Kraak M.-J., Ormeling F.: Cartography, 2nd ed., Prentice Hall, 2003.

Langran G.: Time in Geographic Information Systems, Taylor & Francis, 1993.

OGC Inc.: Simple Features Specification for SQL, Revision 1.1, 1999.

OGC Inc.: Filter Encoding Implementation Specification, Version 1.0.0, 2001.

OGC Inc.: Styled Layer Description Implementation Specification, Version 1.0.0, 2002(a).

OGC Inc.: Web Feature Service Implementation Specification, Version 1.0.0, 2002(b).

OGC Inc.: Style Management Service (SMS), OpenGIS Discussion Paper, Version 0.0.9, 2003(a).

OGC Inc.: Geography Markup Language Implementation Specification, Version 3.00, 2003(b).

Pammer A, Radoczky V.: Multimediale Konzepte für mobile kartenbasierte Fußgängernavigationssysteme, in: Strobl J., Zipf A. (Hrsg.): Geoinformation mobil, Wichmann, pp. 117-126, 2002.

W3C: Cascading Style Sheets, Level 2, CSS 2 Specification, W3C Recommendation, 1998.

W3C: Extensible Stylesheet Language, Version 1.0, W3C Recommendation, 2001.

W3C: Scalable Vector Graphics (SVG) 1.1 Specification, W3C Recommendation, 2003.