EXTENDING OGC STYLED LAYER DESCRIPTOR (SLD) FOR THEMATIC CARTOGRAPHY

Towards the ubiquitous use of advanced mapping functions
through standardized visualization rules

Dietze, Leonhard and Zipf, Alexander

University of Applied Sciences Mainz, University of Bonn
Meckenheimer Allee 172, 53115 Bonn Germany
ph: +49-162 216 4747 / fax: +49-228-73-5607
dietze@geoinform.fh-mainz.de, zipf@geographie.uni-bonn.de

Abstract: Currently we are moving from desktop applications towards the age of ubiquitous cartography. Therefore we have to consider more specialized map functions like thematic cartography. Mobile mapping will not only be limited to navigation systems or typical LBS functions. There are more specialized map functions like thematic mapping that are interesting and will be important in future web-based applications. Therefore a standard covering these aspects is needed in order to allow interoperability. As current specifications only allow basic thematic mappings, we hereby propose an extension to a specification that is already widely used: The OGC Symbology Encoding (SE) as succesor of the SLD specification.

Keywords: cartography, thematic maps, OGC, SLD, symbol encoding (SE)

1. Introduction

Map based applications become ubiquitous (e.g. Morita 2004, Ota 2004, Zipf 2004, UbiGIS). As we are moving from desktop applications towards the age of ubiquitous cartography where people use interactive maps anyplace on a wide variety of devices connected through wireless networks, we have to consider not only maps for navigation purposes, but also more specialized map functions like thematic cartography. Mobile mapping will not only be limited to navigation systems or typical LBS functions like finding the nearest restaurant like it is offered nowadays. There are more specialized map functions like thematic mapping that are interesting and will be important in future web-based applications. Already results from a user survey by the WebPark-project shows that more than 56% of the users would be interested in thematic maps on mobile devices (Diaz et al. 2004).

In order to cope with the heterogeneity of the involved system components these map applications need to be based on interoperable spatial data infrastructures (SDI) using standardized services. The OGC Styled Layer Description (SLD) specification is a formal representation for web-based maps. But SLD still has some limitations (Brinkhoff 2005, Weiser & Zipf 2005. Neubauer & Zipf 2007) - in particular for thematic cartography. Currently only basic thematic maps can be generated: choropleth maps and proportional symbol maps using predefined symbols like e.g. circle, square, cross. Alternatively it is possible to embed pre-compiled external graphics. Therefore it is important to extend this specification in order to allow more advanced symbolization and functions for thematic maps in an explicit way in form of a declarative representation.

2. State of the Art/Standards for thematic mapping

The OGC Styled Layer Descriptor (SLD) (OGC 2005) specification allows to extend the limited possibilities of a WMS-server corresponding the visual representation of the map. A WMS without this would only have the possibility to visualize the map using some pre-defined styles from the server. This can be done more properly using SLD, it allows classifications and this also from the clients´ side.

The SLD-specification is based on an XML-schema that defines the possible elements for the symbolization of the maps. One of the most important element of the SLD is the *rule*-element, as it allows to classify datasets corresponding to some given parameters (using a *filter*). All of the important parameters for the classification have to be set within this element. Sub-elements are e.g. *PointSymbolizer* or *PolygonSymbolizer*, that allow different types of symbolization like fill or stroke.

The OGC Symbology Encoding specification (SE)(MÜLLER 2007) was originally part of the SLD specification. It was then taken out of SLD and made into a separate specification. It allows more possibilities with symbolization than what was available using only SLD. The SE specification only contains the encoding part of SLD without any interfaces so that it could now also be used outside of web services, e.g. in desktop applications. New possibilities with the SE specification compared to SLD include more advanced linear signatures, labeling, better scaling of symbols and others.

The need for the hereby proposed extension of the SE specification was caused through the currently limited possibilities for thematic mapping. Current possibilities in this direction are restricted to choropleth maps and simple point maps. But the current specification is not adequate enough if e.g. pie charts or line charts should be drawn on point maps. It would also be desirable to simplify the classification process for dynamic thematic choropleth maps.

Earlier work in this field by Duarte Teixeira et al. (2005) has been evaluated. In spite of similar ideas we found it not to be fully sufficient for the needs of creating a variety of thematic maps.

3. Types of thematic maps

There are several types of thematic maps that can be used for thematic mappings. But within this paper only some more commonly used types are presented as a base to describe the proposed extension of the Styled Layer Descriptor specification.

3.1. Choropleth maps

These maps are usually used to display or visualize data that comprises of relative values. Polygons are filled entirely to show differences in geographic distribution. Frequently used topics using this type of maps are e.g. population density maps. The dataset has to classified in order to allow the building of classes that can be mapped. This process can be done using one of several methods of data classification like equal intervals, quantiles, standard deviation, maximum breaks, natural breaks, optimal or others (SLOCUM et al. 2004). The class breaks are selected corresponding to the data and to the characteristics that should be visualized using the map. These are also called simple choropleth maps. A example is shown in figure 1.

Derived from the choropleth maps are classless choropleth maps (ROBINSON et al 1995) or so called range maps. They can be used in computer-controlled output devices, as these can easily produce these needed gradations in colour. This was not so easy while targeting mainly on printed maps. This kind of map might seem to be more complex than a simple choropleth map, because it consists of several nuances of colours and this can sometimes not differentiated by the human eye. But on the other hand this type is more logic than the simple one, because it shows the real differences without just generating. An example of a range map can be seen in figure 2 and 8.

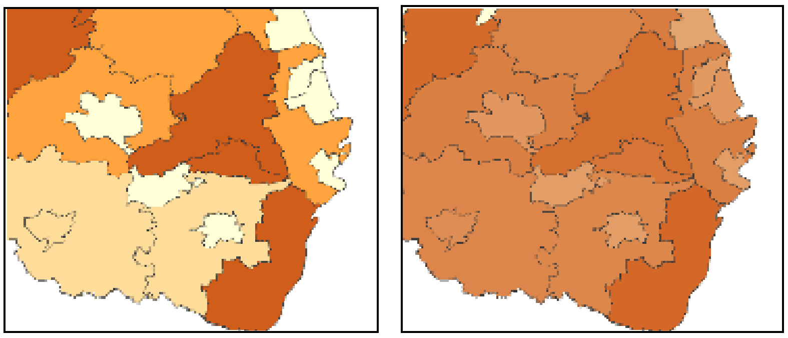


Figure 1 and 2: simple choropleth map (left) vs. range map (right)

3.2 Diagram maps

Diagram maps are thematic maps on which diagrams are placed within a territorial structure. The design and size of these diagrams vary corresponding to some attribute data. The numbers within these maps usually display absolute respectively quantitative values. Since most of the statistical information is based on some kind of administrative unit and not on single points/cities, this method is widely used.

These maps can be used to visualize several different attributes together in one map and thus showing the connection between them. One example could be a map showing the population density of an area together with pie-charts printed on top of them, visualizing another topic like industrial production numbers. This might show that the population density in industrial regions is much higher than in rural areas. An example for a combination between different types and styles of thematic maps can be seen in figure 3.

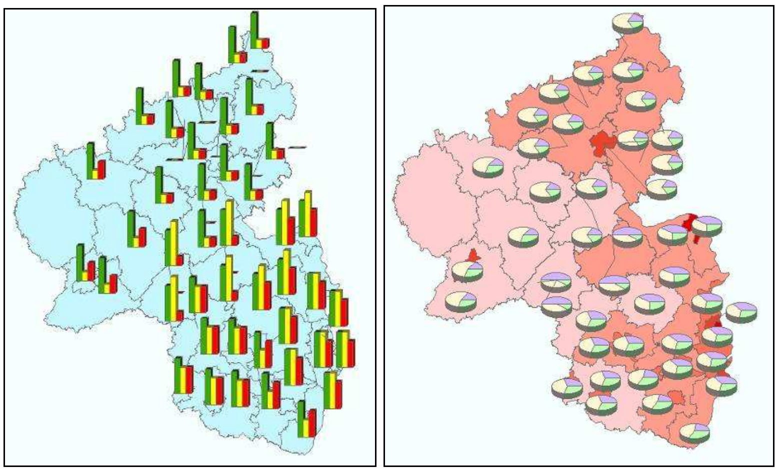


Figure 3: Choropleth map combined with diagram map / pie charts vs. bar charts

4. Thematic Symbology Encoding (TSE) specification

We propose an extension of the current OGC Symbology Encoding (SE) specification as an optional profile of that specification. Similar as the 3D extension to the SE specification by Neubauer & Zipf (2007) we defined an XML–schema based on SE for that purpose. It is called the *Thematic Symbology Encoding* (TSE) schema.

A new namespace was introduced for this extension in order to allow a differentiation between already existing elements or specifications and the new elements. The new namespace is “tse” (xmlns:tse="http://www.i3mainz.de/tse").

4.1 Thematic Symbolizer

All symbolizers of the Symbology-Encoding specification are embedded inside of "*Rules*". They define how a feature should appear on a map. They also describe the graphical aspects of the visualization. Currently there are symbolizers for lines, polygons, points, text and raster. These are now extended through the element "*ThematicSymbolizer*".

A *ThematicSymbolizer* should be used to add advanced cartographic thematic elements to a polygon (or another area-shaped geometry). This polygon might already be filled by the use of a *PolygonSymbolizer* and hereby only a thematic topic is added.

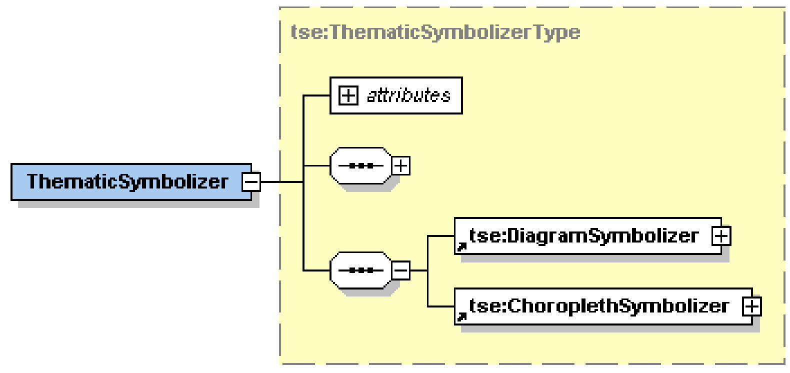


Figure 4: The element *ThematicSymbolizer*

4.2 DiagramSymbolizer

The *DiagramSymbolizer* was introduced to enable to draw point signatures on shapes or surfaces in form of small diagrams or graphs. An example CML-schema diagramm can be seen in Figure 5. Please note: If shapes or polygons are drawn with this *tse:DiagramSymbolizer*, then they are not filled, this "fill" has to be done separately by using the *PolygonSymbolizer* from the SE-specification.

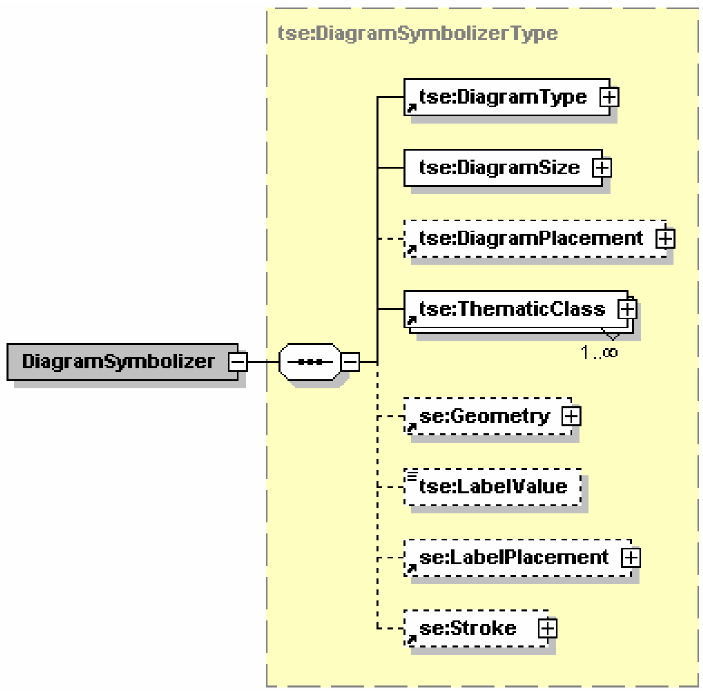


Figure 5: The element *DiagramSymbolizer*

The *DiagramSymbolizer* has the following XML schema definition:

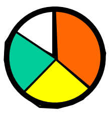
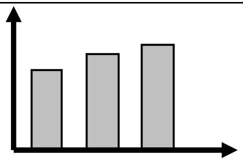
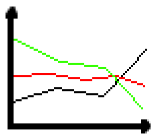
```
<xsd:complexType name="DiagramSymbolizerType">
    <xsd:sequence>
        <xsd:element ref="tse:DiagramType"/>
        <xsd:element ref="tse:DiagramSize"/>
        <xsd:element ref="tse:DiagramPlacement" minOccurs="0"/>
        <xsd:element ref="tse:ThematicClass" maxOccurs="unbounded"/>
        <xsd:element ref="se:Geometry" minOccurs="0"/>
        <xsd:element ref="tse:LabelValue" minOccurs="0"/>
        <xsd:element ref="se:LabelPlacement" minOccurs="0"/>
        <xsd:element ref="se:Stroke" minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>
```

4.2.1 DiagramType

The type of diagram which should be drawn on the given position is specified hereby. It is only allowed to set specific, pre-defined names as *tse:DiagramTypeName*. Possible names are the following: “*barchart*”, “*piechart*” or “*linechart*”. The exact description of the values are then done in the *ThematicClasses*-elements

The following table shows three examples for the above mentioned types of diagrams that should be implemented on the server-side. Further types of diagrams like a “stack chart” are also possible but these have to be implemented and setup individually in the mapservers and called by the use of the *tse:DiagramTypeName.*

Table 1: Diagrams that can be drawn using the DiagramSymbolizer

Name:	**piechart**	**barchart**	**linechart**
Symbol:			

4.2.2 DiagramSize

The size of the diagram is defined using the element *DiagramSize*. It is not optional, one of the following elements has to be set.

The *tse:DiagramRadius* allows to set a consistent radius for all drawn diagrams in the map (e.g. used with pie-charts). Alternatively it is possible to set a fixed width and height (*tse:DiagramHeight*, *tse:DiagramWidth*), this makes sense e.g. for setting the axis of a line or bar chart diagram. The third possibility is to set a dynamic size using the *tse:DiagramSizeValue.* This element can consist of several types of parameters. This enables to set a varying size within a map e.g. for pie charts by using an attribute value.

4.2.3 DiagramPlacement

The element tse:DiagramPlacement was derived from se:PointPlacement of the Symbol Encoding specification. Therefore it is possible to optionally set one or several of the following sub-elements: se:AnchorPoint, se:Displacement, se:Rotation. This element allows to change the positioning of the diagram. With these parameters it is made possible to move it from the given point to another position.

4.2.4 ThematicClass

A *ThematicClass* describes one or more classes (e.g. one slice of a pie, segment, bar, chapter) of a diagram and of its visual appearance. It consists of several compulsive elements. In each *ThematicSymbolizer* a minimum of one *ThematicClass* has to be found, but most of the time there will be more than one. An example-XML document fragment might look like the following:

```
<tse:ThematicClass>
    <tse:ClassLabel>Cars</ tse:ClassLabel>
    <tse:ClassValue>
        <ogc:PropertyName>number_of_cars</ogc:PropertyName>
    </tse:ClassValue>
    <se:Fill>
        <sld:CssParameter name="fill">#FFFFD4</sld:CssParameter>
        <sld:CssParameter name="fill-opacity">1.0</sld:CssParameter>
    </se:Fill>
</tse:ThematicClass>
```

Figure 6 shows the corresponding XML schema-diagram.

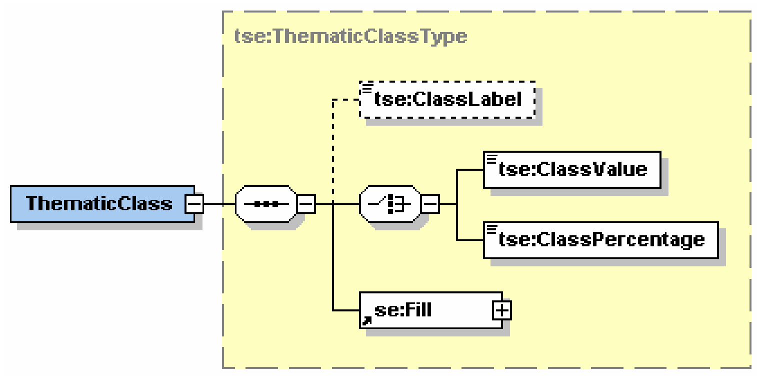


Figure 6: The element *ThematicClass*.

The *tse:ClassLabel* describes the text with which each single class (e.g. slice of a pie chart) should be labeled. This element is optional, if it is not set, then no labeling will be done. Please note that this is not the label of the whole diagram, but the label of each single class.

The value to be drawn in the *ThematicClass* will either be described by the *tse:ClassValue* or the *tse:ClassPercentage*.

The *tse:ClassValue* references an element via the "*ogc:PropertyName*", within which (the attribute) an absolute number is contained. This number should be used for the classification. The sum of all given numbers within all classes is the total number. From this total number in turn it is possible to calculate the angles resp. size of the single segments or classes.

The *tse:ClassPercentage* can be used in alternative to this. It defines the reference to an attribute (by using the *ogc:PropertyName*) with relative percentage values. With these percentages the size of the classes to draw are then calculated and drawn.

The *se:Fill* element has been taken from the SE specification. It allows to do a *GraphicFill* with different parameters and/or *se:SvgParameter*. But please keep in mind that with the given fill-value not the whole diagram is filled but only one single class in form of a bar or pie-slice!

For a more detailed description see SE-specification chapter 11.2.2 or chapter 11.1.3

Each occurring *tse:ThematicClass* defines another part of the diagram to be drawn. If there are several classes, then these classes are drawn corresponding to the given *tse:DiagramType* (see chapter 5.2).

- The number of *ThematicClasses* in the “piechart“ corresponds to the number of slices within a pie chart and *tse:Fill* describes the full of one slice.
- In a “barchart” each *ThematicClass* describes a single bar within this diagram. The color also has to be defined for each single bar.
- Each line or graph is described with the *ThematicClass* when drawing a “linechart”. The *tse:Fill-color* is hereby used as line color.

4.2.5 Geometry

The element *Geometry* is used to find out where the diagram should be positioned. It describes the point-shaped or linear geometry that should be used for the drawing and styling. The element *Geometry* is optional.

The diagram has to be set on the centroid (center of gravity) when a polygon is the base. If this geometry is a linear geometry then the diagram should be set on the middle point (like in the *LineSymbolizer*). When a point geometry is given, then the diagram shall be set on this point. Changes to this position can be done through the element *tse:PointPlacement*.

4.2.6 LabelValue

The element *LabelValue* is used to add textual labels to the diagram. It is optional. There will be no label if it is not set. The element *LabelValue* is of type *ParameterValueType* (from the SE-Spec) and can therefore e.g. contain a reference to an attribute value. The type of the value should be a string, because only this will be drawn (written).

4.2.7 LabelPlacement

LabelPlacement is used to define the positioning of the label relative to the diagram. This element has been taken from the SE-specification. It is also optional. If it describes a point, then it can be moved relatively by using the *se:AnchorPoint*, the *se:Displacement* or the *se:Rotation*. If there is a polygon, then the labeling shall be moved relatively to the starting point (e.g. center of gravity, like described above). When using a line, the middle point is used. Like mentioned in the SE-specification there is the possibility to move the label by using the following elements: *se:PerpendicularOffset, se:IsRepeated, se:InitialGap, se:Gap, se:IsAligned, se:GeneralizeLine*.

4.2.8 Stroke

The element *Stroke* was also taken from the SE-specification. But here it is used to define the type of line that is used to draw the diagrams: It is optional.

4.2.9 Example of a DiagramSymbolizer

The following example shows a possible definition for a thematic map using the *DiagramSymbolizer*.

```
<tse:ThematicSymbolizer>
        <tse:DiagramSymbolizer>
              <tse:DiagramType>piechart</tse:DiagramType>
              <tse:DiagramSize>
                      <tse:DiagramRadius>15</tse:DiagramRadius>
              </tse:DiagramSize>
              <tse:ThematicClass>
                      <tse:ClassValue>
                        <ogc:PropertyName>sold_ handhelds</ogc:PropertyName>
                      </tse:ClassValue>
```

```
                <se:Fill>
                  <sld:CssParameter name="fill">#FFFFD4</sld:CssParameter>
                  <sld:CssParameter name="fill-opacity">1.0</sld:CssParameter>
                </se:Fill>
            </tse:ThematicClass>
            <tse:ThematicClass>
                <tse:ClassValue>
                  <ogc:PropertyName>sold _smartphones</ogc:PropertyName>
                </tse:ClassValue>
                <se:Fill>
                  <sld:CssParameter name="fill">#FECCC4</sld:CssParameter>
                  <sld:CssParameter name="fill-opacity">1.0</sld:CssParameter>
                </se:Fill>
            </tse:ThematicClass>
            <tse:ThematicClass>
                <tse:ClassValue>
                  <ogc:PropertyName>sold_mobilephones</ogc:PropertyName>
                </tse:ClassValue>
                <se:Fill>
                  <sld:CssParameter name="fill">#FD9994</sld:CssParameter>
                  <sld:CssParameter name="fill-opacity">1.0</sld:CssParameter>
                </se:Fill>
            </tse:ThematicClass>
            <se:Geometry>the_geom</se:Geometry>
        </tse:DiagramSymbolizer>
    </tse:ThematicSymbolizer>
```

4.3 ChoroplethSymbolizer

4.3.1 The ChoroplethSymbolizer-element

In contrary to the above described *DiagramSymbolizer*, the *tse:ChoroplethSymbolizer* is a kind of extension to the *PolygonSymbolizer* for thematic cartography. With this element (sub-element of the *tse:ThematicSymbolizer*) advanced choropleth maps can be defined.

Within the scope of the Styled Layer Descriptor Spezifikation (SLD) 1.0 it is possible to draw choropleth maps by using the *ogc:Filter* and several e.g. *ogc:PropertyIsBetween* elements. With several *sld:Rule*'s the classes can be defined and described. But in order to allow this, an exact knowledge of the existing pair of variates is needed and a pre-computed classification has to be done. It is desired to simplify the creation of such map by the use of this symbolizer.

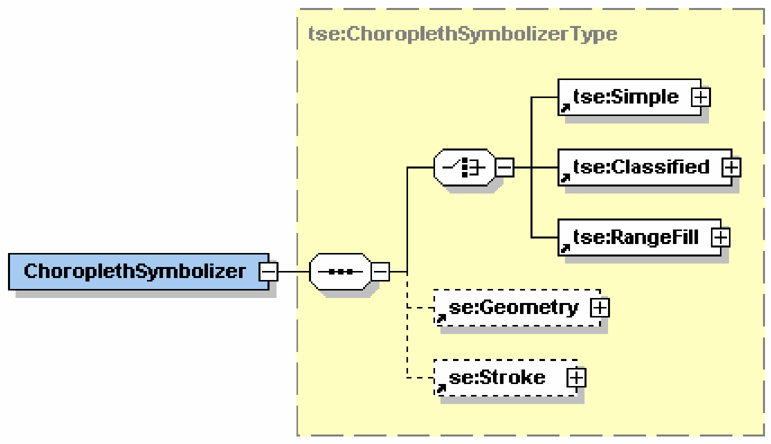


Figure 9: The ChoroplethSymbolizer-element

The *ChoroplethSymbolizer* is defined by the following schema:

```
    <xsd:complexType name="ChoroplethSymbolizerType">
        <xsd:sequence>
            <xsd:choice>
              <xsd:element ref="tse:Simple"/>
              <xsd:element ref="tse:Classified" />
              <xsd:element ref="tse:RangeFill"/>
            </xsd:choice>
            <xsd:element ref="se:Geometry" minOccurs="0"/>
            <xsd:element ref="tse:Stroke" minOccurs="0"/>
        </xsd:sequence>
    </xsd:complexType>
```

The element *se:Geometry* has already been explained. The only difference hereby is the fact, that a reference to an area shaped geometry is expected. This element is also optional, because in most cases there is only one single geometry within a feature.

The element *se:Stroke* has also been taken from the SE-specification. There more details are given. Here it defines the type or style in which the lines of the given area geometry shall be drawn. One of three possible types of classification definitely have to be set: *tse:Simple*, *tse:Classified* or *tse:RangeFill*. These elements are described in the following chapters.

4.3.2 Fill: Simple

A simple fill using the element *tse:Simple* is based on the *se:FillType* as described in the SE-Specification. This fill allows to draw the given geometry using the given parameters *se:GraphicFill* and/or *tse:SvgParameter*. No differentiation or separate classifications are possible.

4.3.3 Fill: Classified

A *Classified*-classification using *tse:Classified* on the contrary to a simple choropleth map allows a simplified categorization of the single classes corresponding to a given named type of classification.

The label *tse:ClassifiedLabel* defines the desired labeling for the polygons or areas in the choropleth map. It consists of a value in the *xsd:string* format.

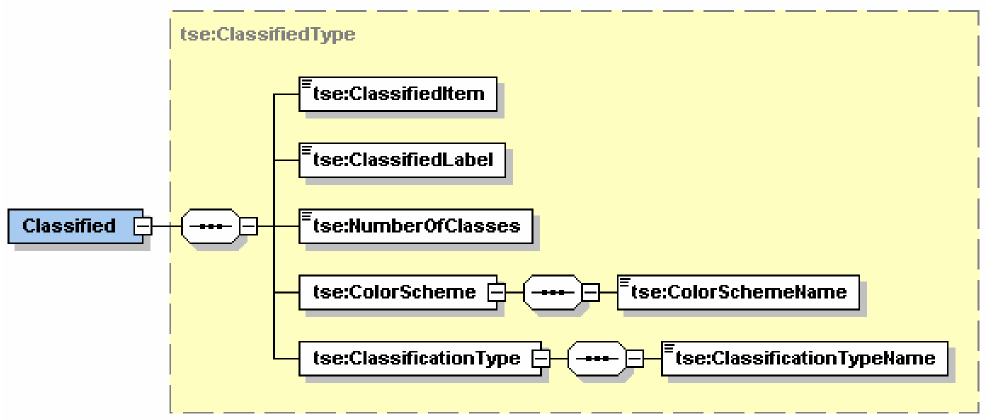


Figure 10: The “Classified”-Element

The *tse:ClassifiedItem* sets the property (*ogc:Property*) within which the values can be found that have to be drawn. There are several possibilities, but the most common method is to directly give a feature using the “*ogc:PropertyName*“. The desired number of classes within this classification is set using the element *tse:NumberOfClasses*. This element is obligatory and is of format *xsd:integer*.

For the *tse:ColorScheme* one of several pre-defined color schemes can be set. This schema has to be set in the sub-element *tse:ColorSchemeName*. Possible identifiers on the server side include among others “*blue2red*“, “*white2red*“, “*white2blue*“, “*white2green*“ etc. Further combinations are also possible if the Mapserver-implementation can provide them. This element names pre-defined classes on the server. The nuances of the colors are then set corresponding to the desired number of classes. A categorization can e.g. be done using the parameters of ColorBrewer (www.colorbrewer.org).

Only pre-defined classification methods can be selected using the *tse:ClassificationType*-element. Its sub-element *tse:ClassificationTypeName* can be set to one of the following classification methods: "*quantile*", "*equal_interval*", "*arithmetic_progression*", "*standard_deviation*" and/or "*natural_breaks*“. With this method, that has to be defined on the server, it is then possible to automatically calculate a classification using the other given parameters. Then the choropleth map can be drawn.

One example of a classified fill is shown below (excerpt from a complete SLD-file):

```
<tse:ThematicSymbolizer>
        <tse:ChoroplethSymbolizer>
             <tse:Classified>
                     <tse:ClassifiedItem>
                       <ogc:PropertyName>pop_density</ogc:PropertyName>
                     </tse:ClassifiedItem>
                     <tse:ClassifiedLabel>
                       <ogc:PropertyName>staat_name</ogc:PropertyName>
                     </tse:ClassifiedLabel>
                     <tse:NumberOfClasses>5</tse:NumberOfClasses>
                     <tse:ColorScheme>
                       <tse:ColorSchemeName>white2red</tse:ColorSchemeName>
                     </tse:ColorScheme>
                     <tse:ClassificationType>
                       <tse:ClassificationTypeName>quantile
                       </tse:ClassificationTypeName>
                     </tse:ClassificationType>
                     <tse:ThematicType>pie</tse:ThematicType>
             </tse:Classified>
             <se:Geometry>staat_geom</se:Geometry>
        </tse:ChoroplethSymbolizer>
</tse:ThematicSymbolizer>
```

4.3.4 Fill: RangeFill

This is a third alternative for defining a classification for a choropleth map. The *tse:RangeFill*-element and its sub-elements allows to visualize a given set of data without knowing and setting the number of classes or single fixed classes. An example for a map using the *RangeFill* can be seen in the following figure 8.

The element *tse:RangeItem* describes (similar to the element *tse:ClassifiedItem* that was already described in chapter 6.3) the property within which the values that have to be visualized can be found. Here it is also useful to set a feature directly using the “*ogc:PropertyName*“.

The *tse:RangeLabel* sets the labeling of the classification in the *xsd:string-format*.

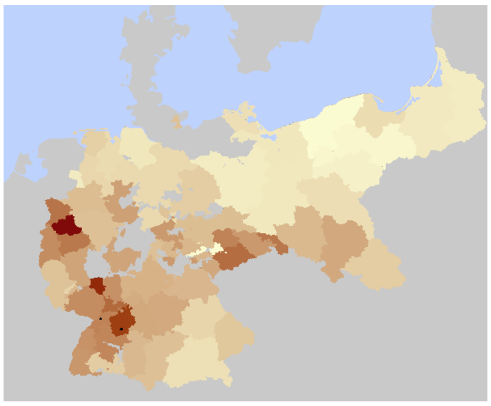


Figure 8: Thematic map using a RangeFill

For setting the element *tse:ColorRange* one of several pre-defined color schemes can be selected. These have to be set in the elements *tse:LowerRGBValue* and *tse:UpperRGBValue*.

The classification within this range-mapping is done as follows: The RGB-value of the element *tse:LowerRGBValue* is defined to visualize the (numerically) lowest value in the given properties (e.g. “#0000FF“). The element *tse:UpperRGBValue* is therefore used to represent the highest (numerical) value (e.g. “#FF0000“).

In the following classification process an automatic continuous classification is applied using these colors. With the element *tse:DataRange* and its sub-elements *tse:LowerBoundary* and *tse:UpperBoundary* the lower and the upper value of the total range of values are set. It therefore also allows to clip extreme values in the upper or lower range (e.g. large cities within a country-wide population density map) in order to allow a meaningful mapping.

5. Future work

It was not possible to include all possibilities for thematic mappings within the scope of this project and paper. Therefore there are still open topics that have to be discussed and that need further deeper research within the field of thematic mapping.

One reasons for this approach of a reduced extension was, above others, the simplicity. It is already quite complex because of the different available options e.g. in the *ChoroplethSymbolizer* using *Simple, Classified* or *RangeFill*. But besides the complexity it is still extendable: further sub-elements can be added within this *ChoroplethSymbolizer* in order to allow other options for the filling or classification process. It is therefore possible to add another element in order to add the functionality for other choropleth-maps to this structure.

Even more possibilities exist for the *DiagramSymbolizer*. Several possible extensions are desirable within the scope of this research project, but it was only possible to narrow the research being done to a limited amount of types of diagrams. Some of possible additional topics within the *DiagramSymbolizer* include - but are not limited to:

- *Stack Chart* - a bar chart where one bar is segmented into several elements. The sizes of them are proportional to some given attribute. The overall size of a bar corresponds to an superior value.
- *Scatterplot* - a chart that displays values for two variables using cartesian coordinates. The data is then visualized as a collection of points.

Future plans include the integration of this extended OGC SE functionality into our current implementation of the OpenLS presentation service (Neis et al 2007) in order to be able to use it within our research projects. Based on this implementation and after further discussions with and feedback from the research community it might be sensible to submit this extension of the symbology encoding specification for consideration as a discussion paper to the Open Geospatial Consortium.

6. References

Brewer: www.colorbrewer.org

Brinkhoff, T. (2005): Towards a Declarative Portrayal and Interaction Model forGIS and LBS. 8th Conference on Geographic Information Science (AGILE 2005), Estoril, Portugal, 2005, pp. 449-458.

Dias, E., Beinat, E., Rhin, C. and Scholten, H. (2004). Location Aware ICT in Addressing Protected Areas' Goals. Research on computing Science vol. 11 (special edition on e-Environment). Edited by Prastacos, P. and Murillo, M. Centre for Computing Research at IPN. Mexico City. p.273-289.

Duarte Teixeira, M., De Melo Cuba, R., Mizuta Weiss, G. (2005): Creating Thematic Maps with OGC Standards through the Web. GML & Geo-Spatial Web Services Conference 2005, Vancouver, Canada.

Morita, T. (2004): Ubiquitous Mapping in Tokyo. ICA UPIMap2004, Tokyo, Japan.

Müller, M. (ed) (2007): OpenGIS Symbology Encoding Implementation Specification v.1.1.0.nr. 05-077r4

Neis, P. and Zipf, A. (2007 accepted): Realizing Focus Maps with Landmarks using OpenLS Services. LBS and Telecartography 2007. Hongkong.

Neubauer, S., Zipf, A. (2007): Suggestions for Extending the OGC Styled Layer Descriptor (SLD) Specification into 3D – Towards Visualization Rules for 3D City Models. Proc. Of. Urban Data Management Symposium. UDMS 2007. Stuttgart. Germany.

OGC: OpenGIS® Styled Layer Descriptor (SLD) Implementation Specification v. 1.0 doc.nr. 02-070

OGC (2005): Styled Layer Descriptor Profile of the Web Map Service Implementation Specification version 1.1 doc.nr. 05-078

Ota, M. (2004): Ubiquitous Path Representation by the Geographic Data Integration. ICA UPIMap2004, Tokyo, Japan.

Robinson, A.H., Morrison, J.L., Muehrcke, P.C., Kimerling A.J., Guptill, S.C. (1995): Elements of Cartography. John Wiley & Sons, Inc, 6th Edition.

Slocum, T.A. (Ed), McMaster, R.B., Kessler, F.C., Howard, H.H. Thematic Cartography and Geographic Visualization. Prentice-Hall 2004, 2nd Edition,

UbiGIS.org – http://www.ubigis.org

Weiser, A., Zipf A. (2005): A visual editor for OGC SLD files for automating the configuration of WMS and mobile map applications. In: 3rd Symposium on Location Based Services and TeleCartography. Vienna. Austria. 2005. Springer.

Zipf, A. (2004): Mobile Anwendungen auf Basis von Geodateninfrastrukturen - von LBS zu UbiGIS. In: Bernard, L.;Fitzke, J.; und Wagner, R. (eds): Geodateninfrastrukturen. Wichmann Verlag. Heidelberg.