A redesign of OGC Symbology Encoding standard for sharing cartography

Erwan Bocher[1] **and Olivier Ertz**[2]

[1]**Lab-STICC CNRS UMR 6285, University of South Brittany, Vannes, Bretagne, France**
[2]**HEIG-VD / Media Engineering Institute, University of Applied Sciences and Arts Western Switzerland, Yverdon-les-Bains, Vaud, Switzerland**

Corresponding author:
Olivier Ertz[2]

Email address: olivier.ertz@heig-vd.ch

ABSTRACT

Despite most Spatial Data Infrastructures are offering service-based visualization of geospatial data, requirements are often at a very basic level leading to poor quality of maps. This is a general observation for any geospatial architecture as soon as open standards as those of the Open Geospatial Consortium (OGC) shall be applied. To improve the situation, this paper does focus on improvements at the portrayal interoperability side by considering standardization aspects. We propose two major redesign recommendations. First to consolidate the cartographic theory at the core of the OGC Symbology Encoding standard. Secondly to build the standard in a modular way so as to be ready to be extended with upcoming future cartographic requirements.
Thus, we start by defining portrayal interoperability by means of typical use cases that frame the concept of sharing cartography. Then we bring to light the strengths and limits of the relevant open standards to consider in this context. Finally we propose a set of recommendations to overcome the limits so as to make these use cases a true reality.
Even if the definition of a cartographic-oriented standard is not able to act as a complete cartographic design framework by itself, we argue that pushing forward the standardization work dedicated to cartography is a way to share and disseminate good practices and finally to improve the quality of the visualizations.

INTRODUCTION

Given how good geospatial technologies take advantage of the constant evolution of information and communication technologies, Spatial Data Infrastructure (SDI) appeared as a new paradigm in geospatial data handling. It extends desktop GIS (Craglia, 2010) where data collected by other organizations can be searched, retrieved and manipulated for several usages (Tóth et al., 2012). Many regional, national and international initiatives have setup well-defined access policies to promote the arrangement of SDI because location information is important in managing everything that a governance has to organize.

Currently, several SDI initiatives are particularly well implemented to encourage data discovery and sharing across different communities with various applications, but SDI users have only access to simple viewing. Despite service-based visualization of geospatial data is part of the SDI components, the shared maps are often of poor quality. Indeed, in the case of the infrastructure for spatial information in Europe, requirements are defined at a very basic level according to INSPIRE Drafting Team (2014), in section 16. Even with the few dedicated recommendations for portrayal rules defined by INSPIRE Drafting Team (2008) in section A.11, the importance of visualization for transforming geospatial data into useful geographical information is still relatively a concern of second zone. Contemporary maps coming from SDI exhibit a serious lack of knowledge in cartography with many map-makers repeating some basic mistakes. Such as maps from Eurostat / Regional Statistics (2017) where population is represented as a choropleth map (e.g. Population on 1st of January in NUTS 2 regions).Field (2014) points out that the current demand is for quantity, not for quality, and it is the Internet (not the discipline of cartography) which is reacting to this demand.

Hopfstock and Grünreich (2009) underline that poor map design results are the consequence of a "too technology- and/or data-driven approach" and propose improvements by making the cartographic design knowledge explicit and operational. Beside such a relevant proposition at the design level, this paper has a focus on the implementation level by making portrayal interoperability operational through the improvement of the open standards dedicated to cartography. Indeed, interoperability is key for SDI as interconnected computing systems that can work together to accomplish a common task. The presence of open standards is required to allow these different systems to communicate with each other without depending on a particular actor (Sykora et al., 2007). The common task presently in question is about the ability for a user community interconnected by interoperable systems to share a cartography used for the authoring of a map. That is, not only the result of a cartographic rendering built of a set of pixels, but also the underlying cartographic instructions which describe how the map is authored. Considering the vision of a SDI as an open participation platform, we can figure out how such an ability would participate to empower all types of users, from the cartographic professional to data artists, journalists and coders (Field, 2014) to gain useful geographical information by means of cartographic visualizations. An ability that contributes to the power of maps as tools which enable the sharing of spatial information and knowledge, but also the collaboration through shared creativity and skills transfer between "produsers" for better decision-making (Bruns, 2013).

For cartographic portrayal interoperability, many SDI policies advise the use of standards from Open Geospatial Consortium (OGC) like the Styled Layer Descriptor (Lupp, 2007) and Symbology Encoding specifications (Müller, 2006). But it seems these standards were not able to bring to reality the above vision that goes as far as considering SDI as open participation platforms. We might blame the facts that the moving from closed monolithic applications to open distributed systems is still undergoing (Sykora et al., 2007) and that cartography must take effect providing a methodology with a user-oriented approach (Hopfstock and Grünreich, 2009). But this paper wants to show how it is also important to have syntactic portrayal interoperability operational with a mature open specification able to standardize the cartographic instructions. We show that the current OGC Symbology Encoding standard does offer limited capabilities for describing cartographic symbolizations. Then, while we develop some recommendations to improve the situation through more capabilities to customize the map symbology, we also propose some good practices to favor the adoption of the standard by implementors so as to make it really operational for long term. We believe that these propositions should lead to rich cartographic portrayal interoperability, going further than basic styles. There is no reason SDI users have to be satisfied with often unsuitable maps.

FROM MAP DESIGN TO PORTRAYAL INTEROPERABILITY

Clearly, many definitions and types of map exist. As Tyner (2010) writes "We all know what a map is, but that definition can vary from person to person and culture to culture". However, many of them do share the idea of a map as an intellectual construction that is based on the experience and knowledge of the cartographer to manipulate data input according initial hypotheses and its capacity to play with graphic signs (Slocum et al., 2009; Tyner, 2010). Furthermore, even if the definition is hard to settle, cartographers have also worked to formalize map syntactics by developing symbol categories and rules to combine them. Visual variables are symbols that can be applied to data in order to reveal information. Largely based on the Bertin and Berg (2010) classification, several cartographic authors agree with a set of commons visual variables (Carpendale, 2003; MacEachren, 2004; Tyner, 2010): shape, size, hue (color), value, texture, orientation (Figure 1).

To create a map, they are individually manipulated or piled up by the cartographer in the process to visually map information about point, line and area features to visual variables (MacEachren, 2004; Slocum et al., 2009). This visual mapping is an embellishment design to improve the aesthetic quality and express efficiently a message (Wood and Fels, 1992). Even if creating map is an aesthetical exercise it's also a science that must respect some rules to make sure that the representation is accurate. A de facto set of best practices based on visual variables has been accepted by the academy of cartographers (Montello, 2002; McMaster and McMaster, 2002). As Bertin and Berg (2010) explains, the choice of the "right" visual variable, which would be most appropriate to represent each aspect of information depends on the type of geographical object but also its characteristics (MacEachren, 2004; Lambert and Zanin, 2013). For example like the statistical nature of the data (qualitative, quantitative) and that raw data must be represented with proportional symbols and a density of values by an areal classification (i.e. a choropleth map).

These map syntactics are the results of the mainstream cartographic theory and the related design knowledge that help to understand how and why certain displays are more successful for spatial inference and decision making than others. This subject is an important issue to improve map quality at the design phase (Hopfstock and Grünreich, 2009). But also at the implementation phase, the theory related to these visual variables to compose map symbols is also suitable to drive the definition of a standardized styling language that must be functionally designed and implemented into the geospatial tools making up SDI.

In order to explain how such a standardized styling language is an essential piece to enable cartographic portrayal interoperability, let's clarify the related concept of sharing cartography. We consider four use cases typical of sharing levels:

- **Level 1: discover**

  At this level, SDI users discover prestyled and ready to be visualized map layers, eventually coming from different systems, they can combine to build a map. For example, it corresponds to the classical geoportal applications offering the user to discover and explore prepared maps and combine prepared layers from various thematics (e.g. map.geo.admin.ch). Typically, it does also match with the story of the fictive SDI user Mr Tüftel in the Web Portrayal Services book (Andrae et al., 2011). Mr Tüftel wants to unify on the same map the water pipes from his municipality but also the pipes from the municipalities in the neighborhood. These are different data sources he wants to combine in his everyday GIS tool. Finally, the user discovers some cartographic facets and gain knowledge of the potential of the underlying data sources hosted by the different systems.

- **Level 2: author**

  Starting from level 1, the potential of the underlying data sources may give to the SDI user some ideas of analytical process which requires to create a new style different from the default. For example, this is useful for Mr Tüftel in the case he would like to create an unified map of water pipes, but with the problem of getting different visualizations of the pipes (e.g. different colors) from the different municipalities. He would then author a common style (e.g. same color) so as to take the control of the whole rendering process. Even further, Mr Tüftel may enrich the analytical process and take benefit of an extra underlying data that classifies each pipe according to its function (either wastewater or rainwater). He would then author a new style (e.g. orange color for wastewater pipes, blue color for rainwater pipes) so as to produce a suitable map to decide where to build the intercommunal water treatment plant.

Starting from level 2 some specific use cases become relevant:

- **Level 3: catalog**

  It is about having at disposal style catalogs offering ready-to-use styles, often tailored for specific thematics, e.g. noise mapping color palettes (EPA, 2011). The ability to import such a specialized symbology into users' tool just avoid to reinvent the wheel in the sense of re-creating the style from scratch. By analogy, the catalog style use case is similar to how the OGC Catalog Service for metadata works.

- **Level 4: collaborate**

  The context of this use case is wider and involves several SDI users into a collaborative authoring process. Several users contribute to the creation of a common map, each user having specialized skills to complement one another so as to tell stories as maps, each using her(his) own software (Ertz et al., 2012). In other words, cartographic portrayal interoperability enable the freedom to the users to work with the tools they are most comfortable and productive with. Also, we may notice the educational capacity of this use case. Considering a team of people with different levels of skills in cartography, there are offered the chance to share them.

As pointed out by Iosifescu-Enescu et al. (2010), "the use of standardized exchange languages is commonly considered as the most practical solution for interoperability especially when it is required to collate resources, like data, from various systems", but also when it is to take the control of a distributed cartographic rendering process. Definitely, starting from level 2, the definition of a standardized styling language is essential to share cartography: that is the underlying cartographic instructions, what we call the symbology code which constitutes a style that describes how a map is authored. Such a definition can be achieved in the same way Iosifescu-Enescu and Hurni (2007) try to define a cartographic ontology by considering that “the building blocks for digital map-making are the primary visual variables (colour, opacity, texture, orientation, arrangement, shape, size, focus) and the patterns (arrangement, texture, and orientation)”. Also, another starting point is to consider a map (either general-purpose maps, special-purpose maps and thematic maps) as being composed of some graphic elements (either geometric primitives or pictorial elements). This approach matches the OGC Symbology Encoding standard which is the standardized styling language (Lupp, 2007) in question here: a style is applied on a dataset to render a map considering a composition of possible symbol elements (called Symbolizer) that carry graphical properties (equivalent to visual variables).

So as to complete the definition of cartographic portrayal interoperability, Figure 2 shows that such a styling language is at the core of the third stage (dedicated to the style rendering) of the cartographic pipeline. Thus it is to notice that the map layout design which configures a title, a legend, a north arrow, a scale bar, etc (Peterson, 2009) is out of our scope, as well as the preprocessing stage which is dedicated to the preparation of the dataset to visualize.

The next part does focus on the technical aspects about how current open standards are able or not to fully meet the conditions of such a cartographic portrayal interoperability.

OPEN STANDARDS FOR SHARING CARTOGRAPHY

Given the concept of sharing cartography defined by the above four use cases, let’s see what are the possibilities and limits to implement them using Open Geospatial Consortium (OGC) standards.

Use case “discover”

The OGC Web Map Service (WMS) standard (De la Beaujardiere, 2006) is currently the only widely accepted open standard for map visualization which standardizes the way for Web clients to request maps with predefined symbolization (Iosifescu-Enescu et al., 2010). This ability, as illustrated with (Figure 3), does match the use case level 1 allowing to discover ready-to-visualize map layers and to combine them to build maps.

Just send a simple GetMap request to the Swisstopo WMS server to get a predefined colored map layer to overlay in your webmapping application (Figure 4):

```
   https://wms.geo.admin.ch/?SERVICE=WMS&VERSION=1.0.0&REQUEST=GetMap&
FORMAT=image/png&LAYERS=ch.swisstopo.pixelkarte-pk25.metadata-kartenblatt&
SRS=EPSG:3857&STYLES=default&WIDTH=2285&HEIGHT=897&BBOX=174582,5648084,
1571851,6196597
```

The WMS GetMap operation allows to choose one of the internal styles prepared by a map-maker (parameter STYLES). Each style is related to one or more datasets attached to the WMS server and ready to be used by an end-user.

Use case “author”

The analysis of the use case level 2 described in chapter 2 shows that it is required to establish an open framework able to facilitate decision making through customized maps. Iosifescu-Enescu (2007) does underline that the WMS standard combined with the Styled Layer Descriptor profile (SLD) and the Symbology Encoding standard (SE) is able to fulfill such a requirement. The ability to drive remotely the authoring of visualizations is the fundamental base for this use case, for example to fulfill the cartographic requirements of Mr Tüftel. He does not want to download the spatial data, he just wants to adjust the visualization according to his specific needs (Figure 5).

Just send the below WMS/SLD request which has a reference to a style file. It includes some SE instructions which allow to get a visualization of the grid of map sheets through a symbology authored by the user client (Figure 6):

```
   https://wms.geo.admin.ch/?SERVICE=WMS&VERSION=1.0.0&REQUEST=GetMap&
FORMAT=image/png&LAYERS=ch.swisstopo.pixelkarte-pk25.metadata-kartenblatt&
```

```
SRS=EPSG:3857&STYLES=&WIDTH=2285&HEIGHT=897&BBOX=174582,5648084,1571851,
6196597&SLD=http://my.server/style.sld
```

The WMS/SLD GetMap operation allows to reference a custom style authored by the user client, either hosted on an external server (parameter SLD) or directly sent with the WMS request (parameter SLD_BODY).

```
<FeatureTypeStyle>
    <Rule>
        <PolygonSymbolizer>
            <Fill>
                <CssParameter name="fill">#FF0000</CssParameter>
            </Fill>
            <Stroke>
                <CssParameter name="stroke">#00FFFF</CssParameter>
                <CssParameter name="stroke-width">1</CssParameter>
            </Stroke>
        </PolygonSymbolizer>
        <TextSymbolizer>
            <Label>
                <fes:PropertyName>Blattnummer</fes:PropertyName>
            </Label>
            <Fill>
                <CssParameter name="fill">#00FFFF</CssParameter>
            </Fill>
        </TextSymbolizer>
    </Rule>
</FeatureTypeStyle>
```

In other words, the user client does take the control of the rendering process that may be distributed among many WMS servers. Indeed, this ability to drive remotely from the user client side (or map viewer) the WMS rendering server does open interesting doors to bring to life the other use cases.

Use case “catalog”

Going further than using a simple WMS GetMap request to get a ready-to-visualize map layer, the deprecated implementation specification (version 1.0, released in 2002) of the WMS/SLD standard (Lalonde, 2002) does offer style management requests like GetStyles. So you get also the underlying symbology instructions of an internal style that has been predefined and used by the server to show a cartographic facet of some spatial data of the underlying datasets. Thus, the retrieved style is ready to be reworked by the user client within a cartographic tool (Figure 7). While such an ability is already interesting for the use case level 2, the SLD 1.0 style management offers not only GetStyles operation but also PutStyles operation. Together, these operations are a good start for the use case level 3 to build a catalog of styles. The WMS service is then the storage point to discover, import and export styles to share with other SDI users through a style catalog.

Nonetheless, it is to notice that the newest SLD 1.1 release does not have anymore the style management requests which is then a step back.

Use case “collaborate”

Finally, for the use case level 4, the Symbology Encoding standard is also a centerpiece (Figure 8). As experimented by Bocher et al. (2012) in the frame of the SOGVILLE/SCAPC2 research projects, SE instructions are encapsulated into a structure of map project that different users share and work together in the frame of a collaborative cartographic authoring process. Indeed, while the OGC OWS Context standard is used to formalize the map project, it does in particular consider SLD and SE to formalize the shared styles used to render the map layers.

Currently, Styled Layer Descriptor SLD 1.0 or Symbology Encoding SE 1.1 (as styling language to formulate symbology instructions) are the more advanced open standards for sharing cartography as illustrated by the above use case levels. These standards are quite largely adopted by server-side rendering systems. It can be explained because SLD is a WMS application profile which is web service oriented. It is the web interface to take control of the rendering engine behind the WMS service (Figure 9). But in 2005, the WMS/SLD 1.1 profile has been released in particular with the aim to extract the symbology instructions into a dedicated standard, the Symbology Encoding standard (SE 1.1). As a consequence, while the SLD profile stays strongly related to WMS service, it is no longer the case for the symbology instructions which can now be used by any styling software component, not only by WMS/SLD.

At the desktop-side there are only few software which correctly and completely implement Symbology Encoding standard together with a graphical user interface to (re)work styles. Indeed, according to Bocher et al. (2011) many implementations have a conformance that is often not fully observed leading to interoperability defects in term of rendering quality. Apart from inherent bugs and dysfunctions of a tool, several reasons can explain this general situation: (1) due to a partial implementation - see ("MapServer") implementation, there are unimplemented symbology instructions, e.g. linejoin and linecap of LineSymbolizer; (2) due to the existence of two versions of symbology instructions between SLD1.0 and SE1.1, these tools may not check this correctly which causes parsing problems of the XML encoding; (3) due to divergent reading of what the standard tries to specify which may result in different graphical visualizations (it means there are ambiguous explanations in the specification) (4) related to the previous point, there is currently no substantial testsuite within the OGC Compliance and Interoperability Testing Initiative ("CITE") to help to disambiguate and test the graphical conformance of an implementation.

While the above arguments do show how it is essential to have a common styling language (currently in the name of OGC SE 1.1), this importance is accentuated by the fact that many changes and proposals have been received by the standard working group, in particular from the scientific community (Duarte Teixeira et al., 2005; Cooper et al., 2005; Sykora et al., 2007; Dietze and Zipf, 2007; Abson and Olivier, 2007; Schnabel and Hurni, 2007; Mays, 2012; Iosifescu-Enescu et al., 2010; Bocher et al., 2011; Rita et al., 2012; Bocher and Ertz, 2015). All these works share a common claim about enhancing SE. It seems the communities of users were frustrated because no substantial new symbology capabilities have been introduced with the release of SE 1.1 except transformations functions. Moreover, Bocher et al. (2011) and Bocher and Ertz (2015), explain that these only new and few capabilities (interpolate, recode, categorize functions) cause confusions and even some regressions.

For instance, despite all the good intentions, there are several limits that come out from the introduction of the categorize function (defined by SE 1.1 standard as the transformation of continuous values to distinct values, e.g. useful to build choropleth maps):

- The definition seems to only match a requirement emphasized by Jenks (Slocum et al., 2009) that classes must cover all the possible values of the data set and must not be discontinuous. However, such a definition has limits considering optimal methods like the Jenks-Fisher classification or Maximum Breaks classifications that may produce intervals with gaps (Slocum et al., 2009) and that it is often better to use the lowest value of the data set as the minimum value of the first interval rather than negative infinity;

- The categorize function is redundant with the concept of Rule of the Symbology Encoding standard. Moreover, it does offer wider possibilities to define precisely value intervals (minimum/maximum values instead of negative/positive infinite, non-contiguous intervals, interval as singleton);

- Similarly, the RasterSymbolizer concept used to control the styling of raster data has been reduced because of the ColorMapEntry concept from SLD 1.0 has been replaced by the categorize transformation function;

- Finally, the introduction of categorize function has also removed from SLD 1.0 the capability to associate a label to an interval when it is an important requirement to have such an information to build a map legend.

Along the same lines, the many proposed extensions of SLD and SE standards have to be analyzed. The purpose is to identify how these cartographic enhancements are relevant for the redesign of the SE standard. By way of other examples, Abson and Olivier (2007) describe four new possibilities to generate thematic maps (CategoryThematicSymbolizer, SimpleThematicSymbolizer, MultiThematicSymbolizer, ChartThematicSymbolizer). A similar approach appears in Dietze and Zipf (2007) (DiagramSymbolizer and ChoroplethSymbolizer) and in Iosifescu-Enescu et al. (2010) to support various diagram types (e.g. pie charts, bar diagrams) to fulfill the complex visualization requirements coming from environmental management.

Also, the specific options introduced within the XSD schemas by some off-the-shelf geospatial software (e.g. “GeoServer”) have to be considered. Of course it is convenient that XML does offer extensibility to add capabilities in the cartographic language and allows at the same time to implement them in the software just like other capabilities, but it may at the same time also create some non-interoperable defects. Clearly, it seems SE1.1 has never been designed with modularization and extensibility in mind and there are no explicit extension points defined in the underlying symbology model. Many proposals coming from research just consider that the extensible nature of XML is not enough. And to have a complete scene of the limits, currently the SE standard does only offer one XML-based encoding and strongly linked to XML modeling principles (Figure 10). As a consequence, it may be difficult for cartographic communities and developers having different encoding preferences (e.g. CSS-like or JSON-based) to get a chance to observe conformance.

To conclude this chapter, while there are clear possibilities to implement the four levels of sharing cartography, it is also clear that a revision of the common styling language played by the SE standard is required. Three major requirements have to be considered:

- Enrich the standard with new cartographic capabilities inline with the evolution of the needs coming from the map-makers community;
- Redesign the underlying symbology model of the standard so as to be modular and extensible for the long-term;
- Consider the possibility to have other encodings than XML.

The next chapter does develop some proposals to fulfill these requirements.

PROPOSALS

The overall purpose is to make these standards more attractive by turning them into “a really useful (cartographic) engine”, quoting the nod to Thomas the Tank Engine alluded by the OGC “Specification Model — A Standard for Modular specifications” document (SWG, 2009), called the modular spec in below.

Before compiling all the Change Requests collected by the SLD/SE Standard Working Group (SWG), one question does arise: how to plug a new requested ability in the standard?. One first and fundamental recommendation is then to consider the modular spec whose release 1.0 has been edited in 2009, at the time the SE standard was already released and thus not in compliance with. Indeed, the modular spec specifies generic rules to organize the internal logical structure of the standard in a modular way so as to strengthen the guarantee of useful and worth standard easy to implement but also to extend.

Modular structure: one symbology core, many symbology extensions

The modular spec fittingly suggests requirements for modularity with the idea of a standard built of one simple core and many extensions which expand the functionality of a specification. Applied to a new revision of the SE standard, the definition of a symbology core requires first to “reverse design” the underlying symbology model of SE1.1. After which, the concrete symbology capabilities have to be extracted and split into many relevant extensions while taking care of the dependencies. The proposed minimal symbology core illustrated by Figure 11 is partially abstract and defined according to the following concepts:

- the Style concept, in charge of the cartographic portrayal of a collection of features stored within a Layer by applying at least one symbology Rule. A feature is described as an abstraction of real world phenomena as defined by GML standard (Portele, 2007);
- the rendering does run feature per feature using a ”one drawing pass” engine;

- each Rule may be scale filtered and does hold at least one Symbolizer;
- each Symbolizer does describe the graphical parameters for drawing the features (visual variables);
- the Style, Rule and Symbolizer concepts hold parameters which are literal values.

Some of the concepts are defined as abstract (in yellow and with italic names in Figure 11) so as to be considered as extension points. Actually, regarding this, we may notice that Craig (2009) does request a similar concept by the use of XML abstract elements which may than be considered as extension points.

Now that the core is ready, some surrounding extensions may be defined so that the engine is really able to perform a rendering. Indeed, alone, the core doesn't concretely "do" anything. As an example, let's introduce the AreaSymbolizer extension which holds a simple and classical symbolizer, call it the AreaSymbolizer concept which describes the graphical parameters for drawing polygonal features with outlined and filled surface areas. The aim of the below explanations is to illustrate with a simple example the extension mechanism and how extension points are expanded.

At first, it is defined that the AreaSymbolizer extension has a dependency with the FeatureTypeStyle extension and the related concepts:

- the portrayal of a Layer built of N instances of GML AbstractFeatureType (Portele, 2007);
- the ability to access features according to Simple Feature SF-2 (Van den Brink et al., 2012);
- the FeatureTypeStyle specialization of the Style core concept;
- the geometry parameter to each Symbolizer extension that depends on this extension (e.g. the below AreaSymbolizer).

Then the geometry parameter is defined with a dependency on the ValueReference extension which hold the ValueReference specialization of the ParameterValue core concept. In a general way, when a parameter has to be assigned with a value, ValueReference does introduce the ability to reference the value extracted from a data attribute of a feature. This is useful when a FeatureType does hold many geometry properties and allows to reference the one to be used by the renderer.

Finally, the AreaSymbolizer extension itself is required, holding the AreaSymbolizer specialization of the Symbolizer core concept. Called PolygonSymbolizer in SE 1.1 and correctly renamed AreaSymbolizer by Craig (2009), it does introduce:

- the symbology ability to draw a surface area according to a filling and an outline;
- the dependency on the FeatureTypeStyle, Fill, Stroke and the Translate extensions;
- the ability to reference the geometry data attribute to be drawn (by means of its dependency on the FeatureTypeStyle extension).

In consequence, an implementation that wants to observe conformance with the AreaSymbolizer extension requires to implement and drive its rendering engine according to all the concepts of the core (thin outline in Figure 12) and the AreaSymbolizer concept with all the other concepts required by dependencies (bold outline in Figure 12).

Nonetheless, even at this point, a rendering engine would neither concretely "do" anything. Indeed, the implementation has then to offer choices related to the filling and the outline. Some more concrete capabilities have to be implemented, for instance with (dashed outline in Figure 12):

- the SolidFill concept, a Fill specialization which introduces the graphical ability to define a solid color value combined with an opacity;
- the PenStroke concept, a Stroke specialization which introduces the graphical ability to draw a continuous or dashed line with or without join and cap;
- the dependent abstract Color concept (and again a concrete choice of color definition has to be done, like with the RGBColor concept which defines a color in the sRGB color space with three integer values).

Having this modularity approach for long term extensibility applied to all the symbolizer concepts, past, present and future, an implementation can with ease manage step by step the evolution of the conformance level of its technical implementation of the standard.

One encoding-neutral conceptual model, many encodings

Currently, SE 1.1 offers a physical model using XML Schema Definition and, at the same time, a natural encoding based on XML. The initial motivation explaining the below recommendation is related to the fact that there is not only XML, but also many other flavors of encoding, JSON-like, CSS-like, YAML-like among many others it is possible to imagine. The important for portrayal interoperability is not the encoding, it is rather the symbology model. That's why the "one encoding-neutral model / many encodings" approach is promising to favor a large adoption of the standard.

This approach has on one side the encoding-neutral model formalized using UML notations, it can be considered as conceptual. With a class diagram, it does describe the portrayal concepts, their relationships, the modular organization, the extension points and the dependencies. We may notice that UML is often prefered when some work is about the design of portrayal concepts. In Zipf (2005), Figure 13 does depict a simplified version of the underlying symbology model as an UML class diagram. Moreover, Craig (2009) does suggest to avoid the XSD attribute concept in the XML encoding so as to be more portable to other structuring languages which do not have the unusual attribute concept of XML Schema, UML in particular. These are more arguments that are in favor of defining at first a conceptual and encoding-neutral model (Figure 13).

On the other side of this recommended approach, doors are open to offer a variety of encodings. Each encoding does translate into a format the UML notations according to mapping rules. At least one default encoding and following the OGC tradition, XML may be this default encoding. It is up to the standard working group to define the mapping rules to translate the semantic of the conceptual model into XML Schema definitions. Indeed, as noticed by Lonjon et al. (2006), the translation from UML to XML requires a thoughtful analysis of the conceptual model so as to define the global mapping rules (e.g. translate a specialization relationship using static or dynamic typing? how to translate a concrete class, an abstract class, the various types of associations? when using attributes or elements? etc). Thus, UML and XML are together a winning combination two times inline with the modular specification which recommend UML "If the organizing mechanism for the data model used in the specification is an object model" and XML "for any specification which has as one of its purposes the introduction of a new XML schema".

Of course, all these questions related to the mapping rules have to be considered for each encoding offered with the standard. We may notice that the OWS Context Standard Working Group adopted a similar approach, offering the default encoding based on XML Atom and planning to provide an OWS Context JSON Encoding soon, according to ("OGC OWS Context").

Style management and parametrized symbolizer

Beyond the tempting recommendation to reintroduce the WMS/SLD GetStyles and PutStyles methods, the management of a catalog of styles has to be expanded. Thus, Craig (2009) does suggest the introduction of a mechanism to reference the definition of a Symbolizer hosted within a catalog. Moreover, the report does enrich the referencing with a symbolizer-parameterization mechanism so as to offer complete symbolizer reusability between different, incompatible feature types. It consists of a list of formal-parameter names and an argument list.

It is to notice that such a mechanism does fit the one specified by (ISO, 2012) in term of parameterized symbol built of dynamic parameters. Thus, in a general way, it is recommended to consider what ISO has already specified concerning the concepts of "collection of symbols and portrayal functions into portrayal catalogue".

Concerning this aspect of style management, the proposal suggests to continue the conceptual work by blending together all these recommendations: reintroduce GetStyles/PutStyles and introduce the mechanism of symbolizer-parameterization inline with ISO (2012).

New symbolizations capabilities

Among the many symbology capabilities that can be extracted from the pending Change Requests at OGC and the research works, we list below (non exhaustively) some relevant ones. Considering the modular structure (see A), each of these capabilities is an extension (e.g. HatchFill is an extension of the Fill abstract concept, just as SolidFill):

- UnitOfMeasure: adds absolute portrayal units of measure (e.g. mm). At least three additional units are added to make measurements more portable between styling representations and rendering environments: portrayal millimeters and portrayal inches as printing measurements, and portrayal (printer's) points commonly used for font sizes.";

- Transformations: allows to perform general affine transformations like Translate, Rotate, Scale, Matrix using homogeneous coordinates on geometries and graphics. Functions: updates the dependency on Filter Encoding (Vretanos, 2010) so as to define symbology-related functions using the mechanism of FE but also to inherit geometry accessors and operators functions (e.g. calculation of a point guaranteed to lie on a surface to create a map with proportional symbols).
- CompoundStroke: allows multiple graphic and/or simpler strokes to be combined together along the linear path. It is interesting to produce complex stroke styles such as rendering a sequence of graphic icons along a line or drawing simple dashed lines between boat-anchor icons;
- CompositeSymbolizer: allows to manage groups of descendant symbolizers as a single unit. It does make the logical grouping more explicit and it allows a group of symbolizers to be remotely referenced;
- HatchFill: adds cross hatching, a method of area filling which is often used and has so simple parameters that it should be established as another filling variety. It is required to allow the configuation of such a filling in a way conventional in cartography, otherwise the user would be forced to emulate cross hatching by fiddling with the GraphicFill concept;
- DiagramSymbolizer: adds diagram symbolization of geographic features, an effective way of visualizing statistical data in a spatial context. It offers support of "Pie", "Bar", "Line", "Area", "Ring", and "Polar" charts in order to allow visualization of multiple data values using diagrams;
- Multiple drawing pass: adds capabilities to order the level of symbol rendering (e.g. to draw nicely connected highway symbols)

REFERENCE IMPLEMENTATION

The OrbisGIS platform has been used to prototype an implementation of the symbology model all along the standardization work by iterations with tests and validations. At long term, this platform may be adopted as a reference implementation at the OGC ("Compliance and Interoperability Testing Initiative").

OrbisGIS is a Geographical Information System designed by and for research (Bocher and Petit, 2013) which is the main advantage for research communities comparing to other GIS. Indeed, OrbisGIS doesn't intend to reproduce classical GIS functionalities. It is designed to explore new issues or questions in the field of geospatial techniques and methods (such as language issues to query spatial informations and issues on cartography about standardization, semantics and user interface design. To address these challenges, the OrbisGIS architecture (object and data model) and its user interface are frequently redesigned. This approach is fundamental to test the concepts and the ideas related to the ongoing standardization process of symbology standards at OGC. Furthermore, the fact that we have a common set of GIS features organized with the dynamic module system OSGi to access to the geodata, library to use simple features functions, layer model, rendering engine, etc (OSGi, 2014) gives flexibility to plug some experimental code without breaking the platform and the user can easily switch from one to another plugin (Figure 14). More importantly, the usage of OSGi technology does offer a way to implement the modularization principles depicted in the above (i.e. one OSGi bundle per symbology extension).

Another motivation is related to the license. OrbisGIS is an open source software, distributed under the GPL3 license and therefore grants four freedoms (1) to run the program for any purpose, (2) to study how the program works and adapt it to your needs, (3) to redistribute copies so you can help your neighbour, (4) to improve the program, and to release your improvements to the public, so that the whole community benefits (Steiniger and Hunter, 2012).

This aspect is essential in order to have a reference implementation available for the community of implementers of a standard, guiding them better in the understanding of a specification. Given the core principle of science that having open source code available does enable reproducibility (Ertz et al., 2014), we argue that this is also valid for open standards. At one side, it is easy for other researchers and businesses to verify and re-use new developments and adapt them to their needs (Steiniger and Hunter, 2012). Furthermore, having the code of the rendering engine, the user interfaces and all the tests fully accessible should facilitate the understanding and the dissemination of standards for portrayal interoperability while minimizing interoperability defects. In the following we describe the main aspects covered by OrbisGIS to implement the proposed redesign of the symbology model.

XML encoding/decoding

In the context of a prototyping iteration, the symbology model presented in the chapter 4 has been transposed to a XSD schema ("orbisgis/ogc-custom-jaxb"). The Java Architecture for XML Binding (JAXB) library is used to generate the XSD schema-derived Java binding classes. Finally, a Java Style Object Model is built. Thus, symbology instructions are stored in a style file using XML encoding and is parsed prior to be applied by the rendering engine.

Rendering engine

The rendering engine is a OSGi bundle whose mechanism is divided into 12 sequences (Figure 15):
(1) User interface event to draw a map.
(2) The renderer engine gets the style file that contains the symbology instructions.
(3, 4 and 5) The style file is read by the XML parser to create the Java Style Object Model composed of rules and symbols.
(6) The renderer engine starts to draw the style object looping over each rules.
(7) Each rule is scanned to check if a filter must be applied. The filter condition (e.g. select all values greater than…) is prepared for each symbolizer of the rule.
(8) The renderer engine starts to draw all symbols available in the Java Style Object Model. (9) Each symbol reads the data source on which the style must be applied.
(10) A set of features according to the potential filter constraint of the symbolizer is returned (including geometries and data attributes).
(11) The symbols are filled with the features properties to create the graphic elements and visual variables.
(12) Finally, the renderer engine displays the style as a map image.

User interfaces

OrbisGIS offers two user interfaces for configuring the map styles using the capabilities of the underlying symbology model:

- The first one is a productivity tool organized around a set of widgets each dedicated to a common thematic map (Figure 16). The possibilities are limited to what these widgets are able to configure related to what they have been built for. Nonetheless, the second tool can then be used in an expert mode to go further.
- The second one is rather intended for an expert who want to tinker and tweak (Figure 17). The second one is rather intended for an expert who want to tinker and tweak (Figure 17). As an advanced style editor, it is a flexibility tool which allows to manipulate all elements of the symbology model (Rule, Symbols, visual variables). A good knowledge of the symbology model is required because each elements of the style must be set individually. Consequently, the user can express without any limitation (except the limits of the symbology model itself) all her(his) creativity to build complex and outstanding cartographic visualizations.

To illustrate some results rendered with OrbisGIS we present two maps extracted from http://se.orbisgis.org/.

The first one shows a bivariate map to display the number of building permits in Europe in 2005 compared to 2014 (Figure 18).

Bivariate map is a common technique to combine visual variables. The map uses the same type of visual variable to represent two values (as half circles). The main symbology elements used to create this bivariate map are:

- The style element contains 2 rules named A and B;
- Rule A contains one symbolizer element (AreaSymbolizer) to display the stroke of the european countries;

- Rule B defines the bivariate proportional symbol with two elements of PointSymbolizer (for readability, we present only the left half-circle variable);
- The PointSymbolizer contains several sub-elements :
  - the geometry element allows specifying which geometry attribute is to be rendered;
  - the ST_PointOnSurface is an OGC filter function (Vretanos, 2014) used to have a point geometry guaranteed to lie on the surface. This new point derived from the input geometry is the location where to anchor a MarkGraphic, otherwise the symbol might be applied on all the vertices of a geometry;
- The MarkGraphic is defined by :
  - the symbol shape identified by a well-known name, HALFCIRCLE (right side);
  - the size of the shape varies according the height of its view box;
  - to have the shape size proportional with the number of building permits in 2015:
    - an interpolate function is applied on;
    - it uses a ValueReference that points to the attribute named permits2005;
    - the interpolation is defined by two interpolation points chosen along a desired mapping curve (here the minimum and maximum values);
    - for each interpolation point the height of the view box is specified with a specific unit of measure;
  - because the half-circle shape is drawn to the right side, a 180-degree rotation is operated;
  - to finish, the MarkGraphic is filled with a RGB color.

The second map shows a combination of several visual variables: shape, size, color, patterns and orientation (Figure 19). The style is organized around 6 filtered rules that correspond to the biogeographic regions in Switzerland. We present two Rules (A and B) that use the HatchFill and GraphicFill concepts which are extensions of the Fill abstract concept of the symbolizer model.

CONCLUSION

Considering the fundamental works of Bertin and Berg (2010) and successors, the community of map makers has constantly investigated questions about cartographic visualisations in term of design using the appropriate visual variables and combining them together with relevancy. Despite an important body of principles and practices, the community did not grasp the questions about standardization. However, given the multiplicity of software used to flood the world with maps, these questions are nowadays a strategic challenge to be considered in relation with operational requirements.

Even if the definition of a cartographic-oriented standard is not able to act as a complete cartographic design framework by itself, we argue that pushing forward the work aiming at the creation of dedicated standards for cartography is a way to share and disseminate good practices. Indeed, too much spatial data infrastructures do merely accept the limits of the current standards and consequently poor map design. While they have to apply them, it is essential to build them so as to be able to enrich their cartographic capabilities at long-term, to make grow up the good practices and finally to improve the quality of the visualizations. In this sense, we have identified some use cases showing how it is important to make portrayal interoperability operational for sharing cartography, from discovery to collaboration activities, by way of authoring and cataloging activities.

From research results in link with the dedicated SLD/SE OGC Standard Working Group (Ertz and Bocher, 2010), this paper does extract some recommendations to enable portrayal interoperability. They invite to improve the OGC Symbology Encoding standard based on principles and practices in cartography. We start from a functional definition of a map translated into a set of visual variables which are combined to create symbols and finally a map style. The proposed recommendations do observe this functional definition which is at the heart of how SE standard has been specified by OGC.

The design approach is driven by a conceptual definition of the model and unconstrained by specific encoding aspects. As soon as the model is ready, then a default encoding is offered (e.g. XSD/XML). Follow on from this approach of dissociation, it does allow the definition of other encodings according to the various flavors within the communities.

Given that the cartographic requirements will progress during time due to practices growing up and according to domain specific features, the offered symbology model is empowered so as to be extensible and ready to offer new cartographic methods. Moreover, such a modular approach allows implementations to be compliant step-by-step. As a consequence the adoption of the standard should be favored.

Finally, we claim to a testsuite within the OGC Compliance and Interoperability Testing Initiative so as to help to disambiguate and test the visual conformance of the implementations. While it shall be associated to reference implementations, having at least one opensource is also essential for the community of implementers, guiding them even more in the understanding of the standard. In this sense, OrbisGIS is a platform that has been used to prototype an implementation of the symbology model all along the standardization process by iterations with tests and validations. It may become an opensource reference implementation.

REFERENCES

Abson, S.-T. and Olivier, E. (2007). Towards web services dedicated to thematic mapping. osgeo journal. *OSGeo Journal*, 3:30–34.

Andrae, C., Graul, C., Over, M., and Zipf, A., editors (2011). *Web portrayal services: OpenGIS web map service, styled layer descriptor, symbology encoding und ISO 19117 portrayal vorgestellt und erläutert*. OpenGIS essentials : die Geo-Standards von OGC und ISO im Überblick. Wichmann, Berlin. OCLC: 846108747.

Bertin, J. and Berg, W. J. (2010). *Semiology of graphics: diagrams, networks, maps*. ESRI Press : Distributed by Ingram Publisher Services, Redlands, Calif, 1st ed edition.

Bocher, E. and Ertz, O. (2015). Towards Cartographic Portrayal Interoperability – the Revision of OGC Symbology Encoding Standard. In *Proceedings of the 1st ICA European Symposium on Cartography*, volume 1, pages 116–119.

Bocher, E., Ertz, O., Laurent, M., Hégron, G., Petit, G., and Rappo, D. (2011). Cartographie et standard : du modèle a l'utilisateur. In *Proceedings of the 25th International Cartographic Conference Paris, 3-8 July 2011.*, Paris. ICC. OCLC: 781048687.

Bocher, E. and Petit, G. (2013). ORBISGIS: Geographical Information System Designed by and for Research. In Bucher, B. and Ber, F. L., editors, *Innovative Software Development in GIS*, pages 23–66. John Wiley & Sons, Inc.

Bocher, E., Petit, G., Gueganno, A., Fortin, N., Gourlay, A., and Ertz, O. (2012). Séminaire de restitution du SOGVILLE (Système d'Observation Géographique de la Ville). Technical report.

Bruns, A. (2013). *From Prosumption to Produsage*, pages 67–78. Edward Elgar, Cheltenham, UK.

Carpendale, M. S. T. (2003). Considering visual variables as a basis for information visualisation. Technical report, University of Calgary, Calgary, AB.

Cooper, M., Sykora, P., and Hurni, L. (2005). The Role of Cartography within Distributed Software Systems: What can we Contribute? How can we Prosper? In Lapaine, M. and Association, I. C., editors, *22nd International Cartographic Conference and 13th General assembly of the ICA*, A Coruña, Spain. International Cartographic Society.

Craglia, M. (2010). Building inspire: The spatial data infrastructure for europe. *ArcNews*, 32:1–6.

Craig, B. (2009). Ogc ® ows-6 symbology encoding (se) changes (ogc 09-016). Technical report, Open Geospatial Consortium, Inc., Wayland, Massachusetts.

De la Beaujardiere, J. (2006). OpenGIS Web Map Server Implementation Specification (OGC 06–042 Version 1.3. 0). *Open Geospatial Consortium Inc., USA, 85pp*.

Dietze, L. and Zipf, A. (2007). Extending OGC Styled Layer Descriptor (SLD) for Thematic Cartography - Towards the ubiquitous use of advanced mapping functions through standardized visualization rules. In *4th Int. Symp. on LBS and Telecartography*.

Duarte Teixeira, M., De Melo Cuba, R., and Mizuta Weiss, G. (2005). Creating thematic maps with ogc standards through the web. gml and geo-spatial web services. Vancouver, British Columbia.

EPA (2011). Guidance note for strategic noise mapping : Guidance note for strategic noise mapping for the environmental noise regulations 2006. Technical report, Environmental Protection Agency, Wexford, Ireland.

Ertz, O. and Bocher, E. (2010). Styled layer descriptor and symbology encoding 1.2 swg. Technical report, Open Geospatial Consortium, Inc., Wayland, Massachusetts.

Ertz, O., Julien, L. G., and Bocher, E. (2012). Collaborative authoring and polypublication of cartographic content. In *OGRS2012 Symposium Proceedings*, Yverdon Les Bains. lulu.com.

Ertz, O., Rey, S. J., and Joost, S. (2014). The open source dynamics in geospatial research and education. *Journal of Spatial Information Science*, (8).

Eurostat / Regional Statistics (2017). Available at http://ec.europa.eu/eurostat/ (accessed july 6, 2016).

Field, K. (2014). A cacophony of cartography. *The Cartographic Journal*, 51(1):1–10.

Hopfstock, A. and Grünreich, D. (2009). A user-oriented map design in the sdi environment using the example of a european reference map at medium scale. In *Proceedings of the 24th International Cartographic Conference*, Santiago de Chile, Chile. ICC.

INSPIRE Drafting Team (2008). D2.6: Methodology for the development of data specifications.

INSPIRE Drafting Team (2014). D2.5: Generic conceptual model.

Iosifescu-Enescu, I. (2007). Se implementation specification change request – extensions for thematic mapping (ogc cr07-105). Technical report, Open Geospatial Consortium, Inc., Wayland, Massachusetts.

Iosifescu-Enescu, I., Hugentobler, M., and Hurni, L. (2010). Web cartography with open standards – A solution to cartographic challenges of environmental management. *Environmental Modelling & Software*, 25(9):988–999.

Iosifescu-Enescu, I. and Hurni, L. (2007). Towards cartographic ontologies or how computers learn cartography. In *Proceedings of the 23th International Cartographic Conference*, Moscow, Russia. ICC.

ISO (2012). Iso 19117:2012 - geographic information – portrayal.

Lalonde, W. (2002). Styled layer descriptor profile of the web map service implementation specification (ogc 02-070). Technical report, Open Geospatial Consortium Inc, Wayland, Massachusetts.

Lambert, N. and Zanin, C. (2013). Espon cartographic language - mapping guide. Technical report, UMS RIATE, Université Paris Diderot, Paris.

Lonjon, A., Thomasson, J.-J., and Maesano, L. (2006). *Modélisation XML*. Architecte logiciel. Eyrolles, eyrolles edition.

Lupp, M. (2007). *Styled Layer Descriptor profile of the Web Map Service Implementation Specification (OGC 05-078r4)*. Open Geospatial Consortium Inc, Wayland, Massachusetts.

MacEachren, A. M. (2004). *How Maps Work - Representation, Visualization, and Design*. Guilford Press.

Mays, J. (2012). Using sld definitions to display charts in a deegree wms. Cape Town. FOSS4G 2008.

McMaster, R. and McMaster, S. (2002). A History of Twentieth-Century American Academic Cartography. *Cartography and Geographic Information Science*, 29(3):305–321.

Montello, D. R. (2002). Cognitive Map-Design Research in the Twentieth Century: Theoretical and Empirical Approaches. *Cartography and Geographic Information Science*, 29(3):283–304.

Müller, M. (2006). *Styled Layer Descriptor profile of the Web Map Service Implementation Specification (OGC 05-078r4)*. Open Geospatial Consortium Inc, Wayland, Massachusetts.

OSGi (2014). The osgi alliance osgi core. Technical report, OSGi Alliance, San Ramon, USA.

Peterson, G. N. (2009). *GIS Cartography: A Guide to Effective Map Design*. CRC Press, 1 edition. ISBN: 978-1-4200-8213-5.

Portele, C. (2007). Opengis® geography markup language (gml) encoding standard, version 3.2.1 (ogc 07-036). Technical report, Open Geospatial Consortium Inc, Wayland, Massachusetts.

Rita, E., Borbinha, J., and Martins, B. (2012). Extending sld and se for cartograms. In *GSDI 12 World Conference*, Singapor. Global Spatial Data Infrastructure Association.

Schnabel, O. and Hurni, L. (2007). Diagram markup language - a new model for symbolization in internet maps. In *Proceedings of the 23th International Cartographic Conference*, Moscow, Russia. ICC.

Slocum, T. A., MacMaster, R. B., Kessler, F. C., and Howard, H. H. (2009). *Thematic cartography and geovisualization, 3rd Edition*. Pearson Education.

Steiniger, S. and Hunter, A. J. S. (2012). *Free and Open Source GIS Software for Building a Spatial Data Infrastructure*, pages 247–261. Springer Berlin Heidelberg, Berlin, Heidelberg.

SWG, P. (2009). The specification model — a standard for modular specifications (ogc 08-131r3). Technical report, Open Geospatial Consortium Inc, Wayland, Massachusetts.

Sykora, P., Schnabel, O., Enescu, I. I., and Hurni, L. (2007). Extended Cartographic Interfaces for Open Distributed Processing. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 42(3):209–218.

Tyner, J. A. (2010). *Principles of map design*. Guilford Press, New York. OCLC: 699813390.

Tóth, K., Portele, C., Illert, A., Lutz, M., and Nunes de Lima, M. (2012). *A Conceptual Model for Developing Interoperability Specifications in Spatial Data Infrastructures*. Publications Office, 2012, Luxembourg.

Van den Brink, L., Portele, C., and Vretanos, Panagiotis (Peter), A. (2012). Geography markup language (gml) simple features profile - with corrigendum (ogc 10-100r3). Technical report, Open Geospatial Consortium Inc, Wayland, Massachusetts.

Vretanos, P. (2010). Opengis filter encoding 2.0 encoding standard (ogc 09-026r1 and iso/dis 19143). Technical report, Open Geospatial Consortium Inc, Wayland, Massachusetts.

Wood, D. and Fels, J. (1992). *The power of maps*. Mappings. Guilford Press, New York. OCLC: 26399801.

Zipf, A. (2005). *Using Styled Layer Descriptor (SLD) for the Dynamic Generation of User- and Context-Adaptive Mobile Maps – A Technical Framework*, pages 183–193. Springer Berlin Heidelberg, Berlin, Heidelberg.

730 **FIGURES**

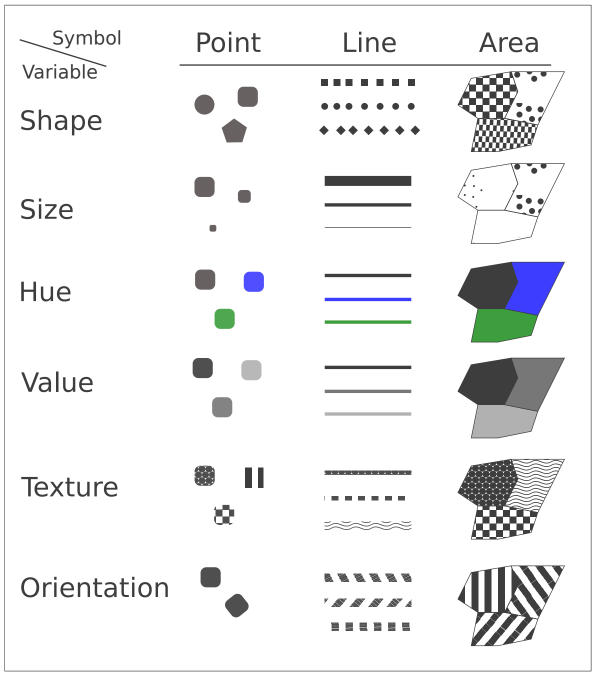


Figure 1. The visual variables of symbols.

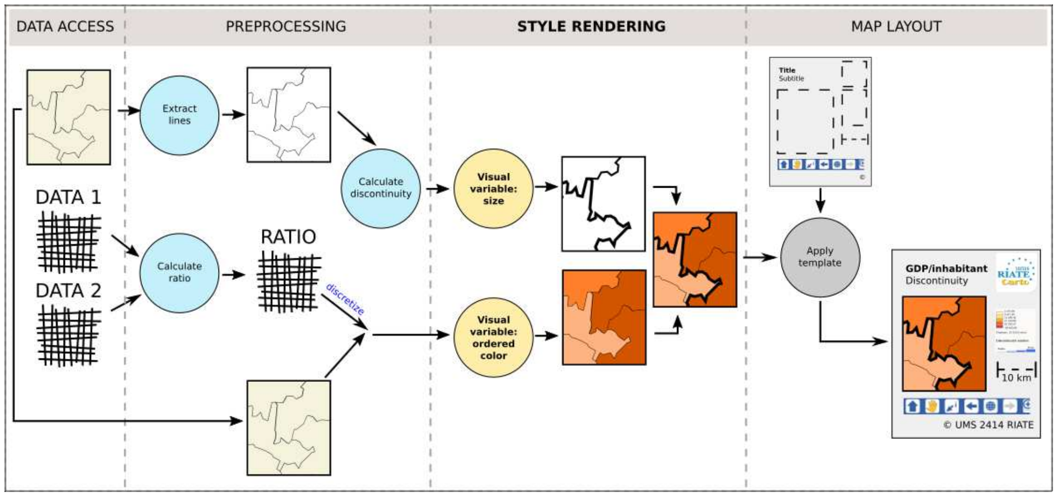


Figure 2. The four stages of the cartographic map design, inspired from (Lambert and Zanin, 2013).

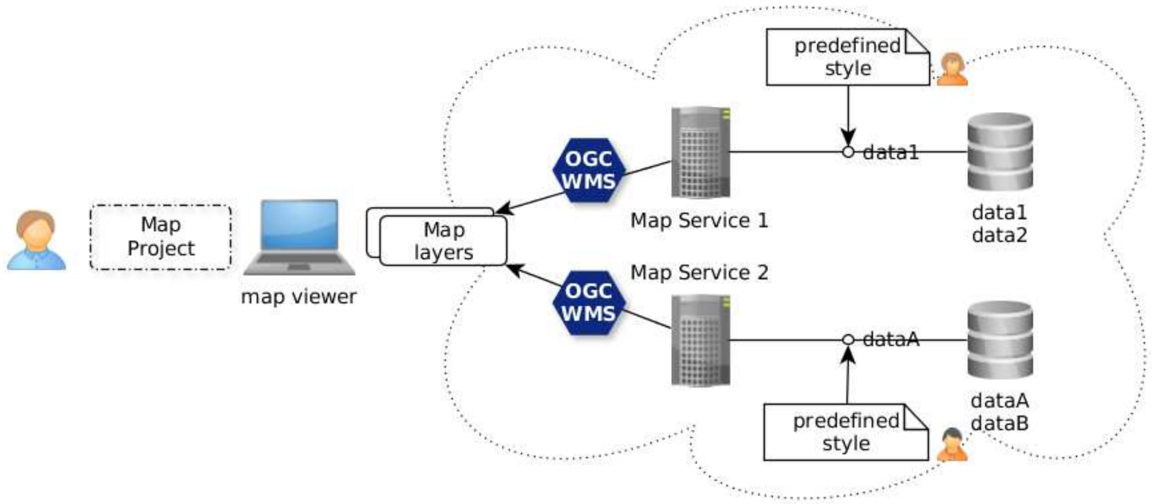


Figure 3. Discover ready to be visualized map layers with OGC WMS standard.

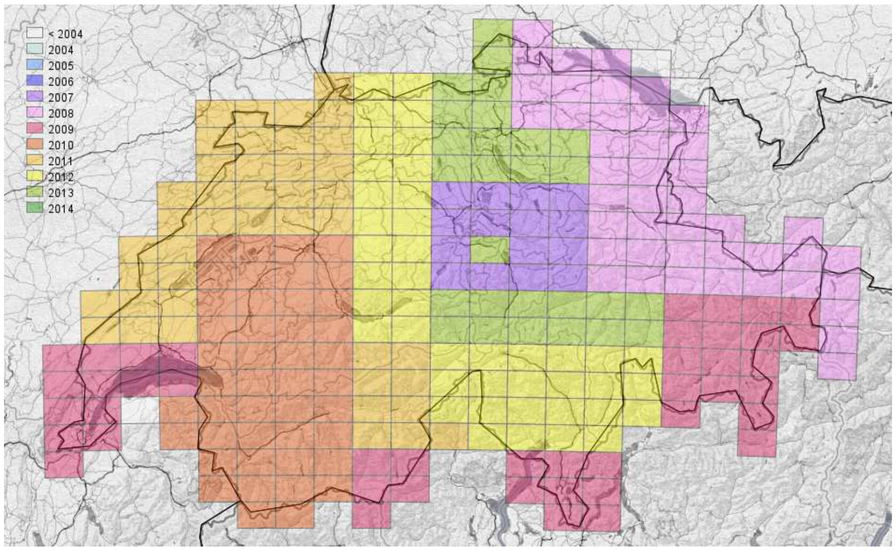


Figure 4. Visualization of the grid of map sheets of Switzerland (1:25000) through a default cartographic style showing a choropleth symbology based on the year of edition of the sheet.

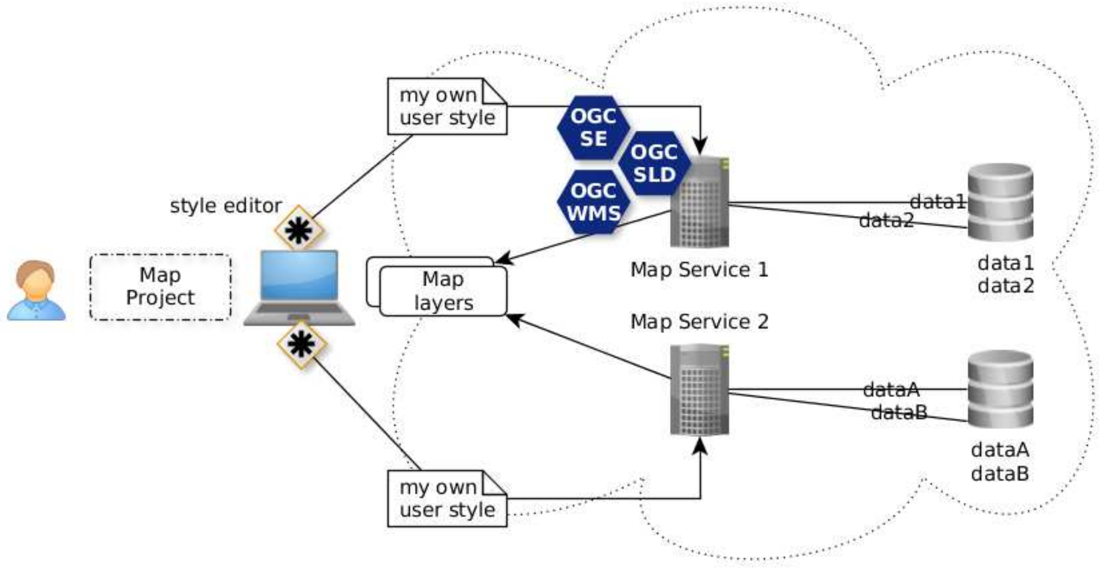


Figure 5. Authoring of user style to visualize map layers with OGC WMS/SLD and SE standards.

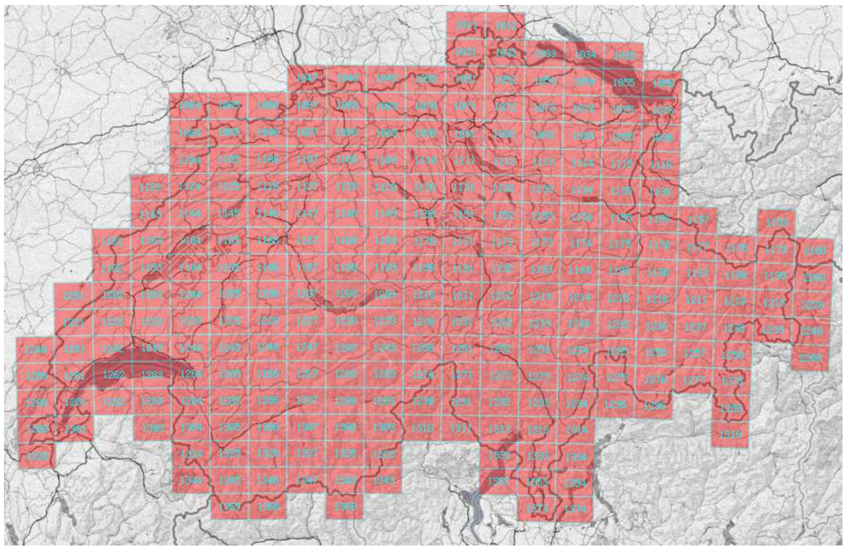


Figure 6. Visualization of the grid of map sheets of Switzerland (1:25000) through another cartographic facet showing labels based on the sheet number.

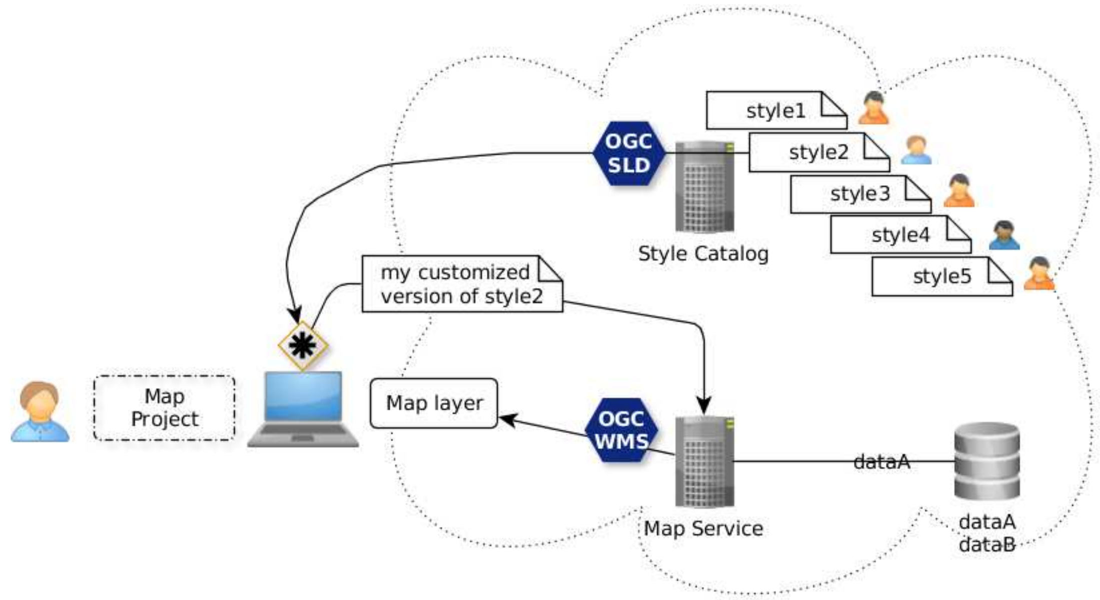


Figure 7. Re-authoring of styles shared by catalogs with OGC WMS/SLD standards.

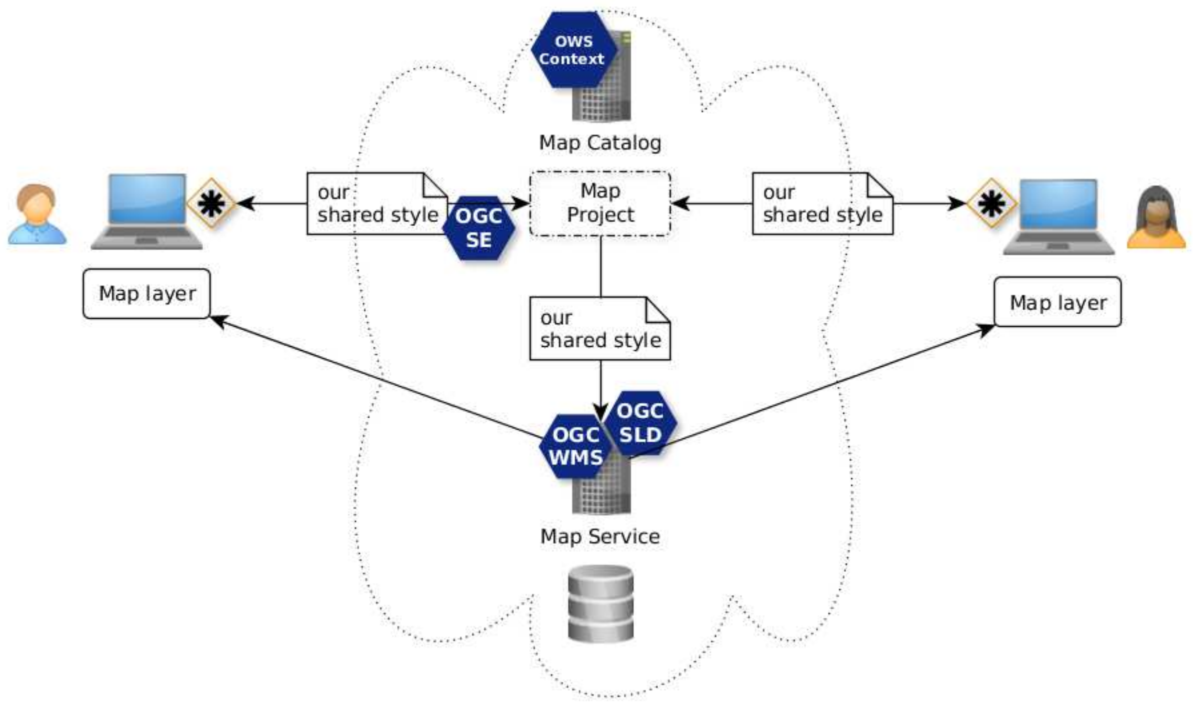


Figure 8. Creation of a common map based on shared styles with OGC WMS/SLD, SE and OWS Context standards.

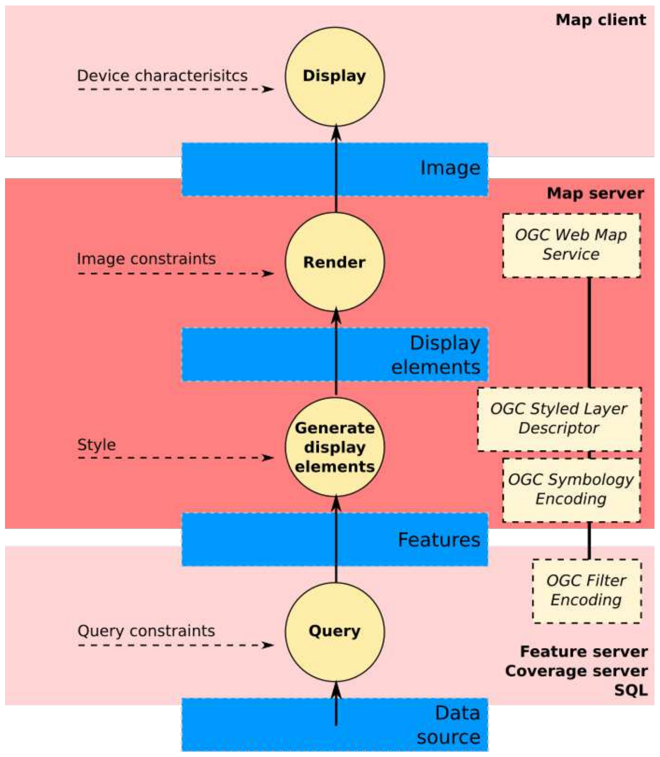


Figure 9. OGC portrayal model.

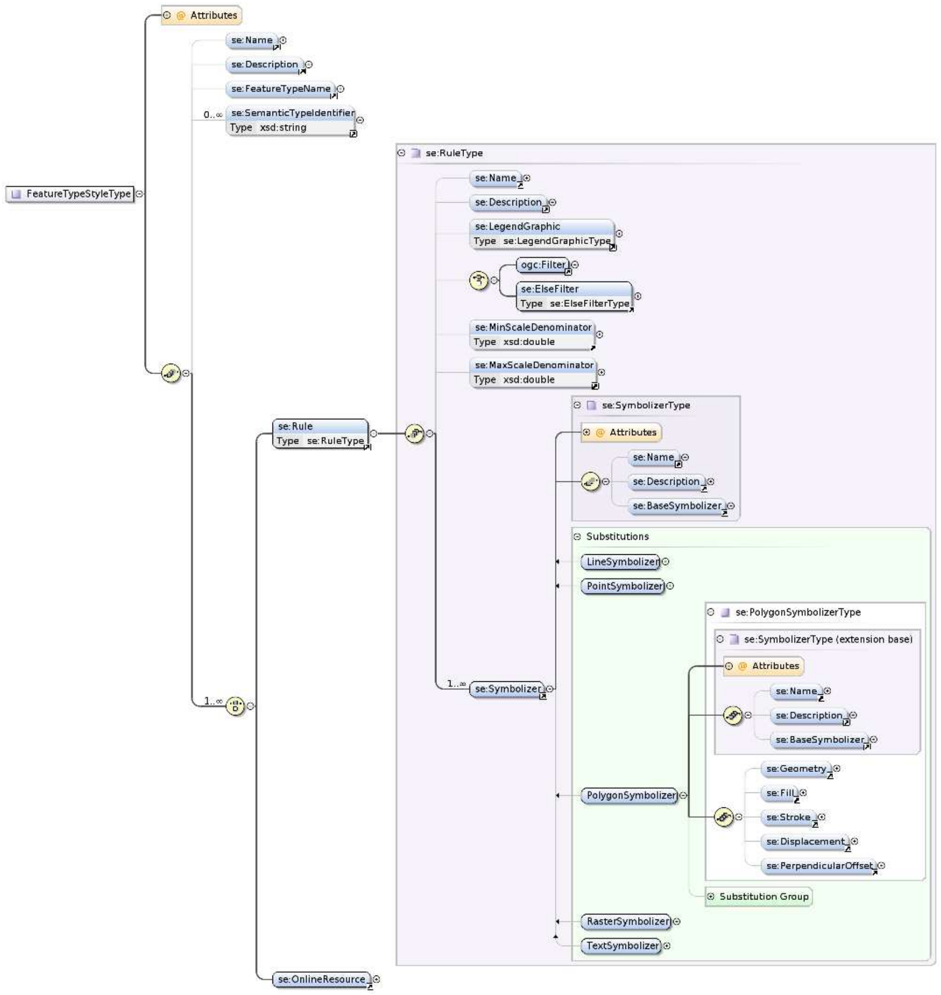


Figure 10. The physical symbology model of SE formalized with XSD, see also ("Schema documentation for FeatureStyle.xsd").

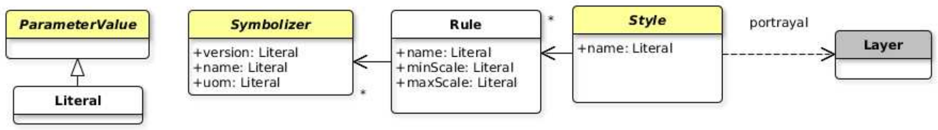


Figure 11. Recommendation for a minimal symbology core.

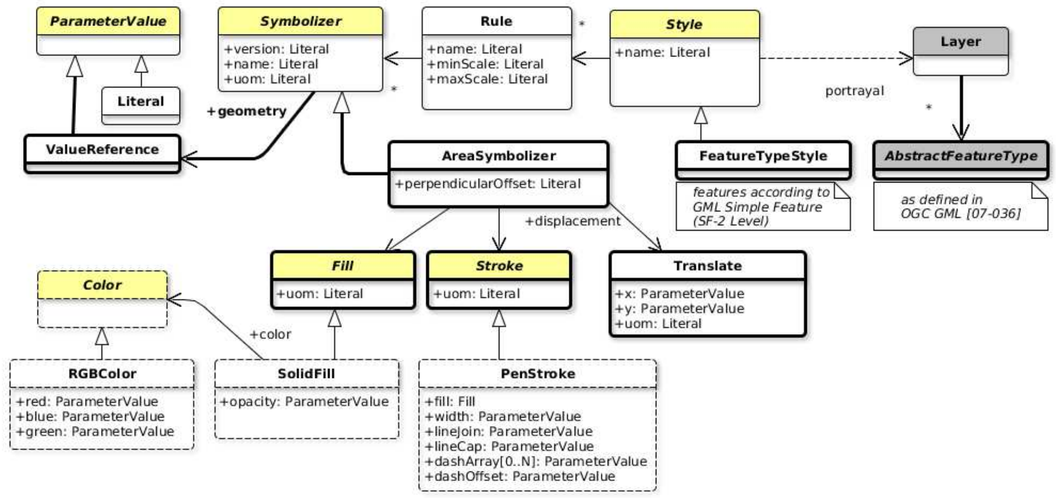


Figure 12. Concepts to implement so as to observe conformance with the AreaSymbolizer extension.

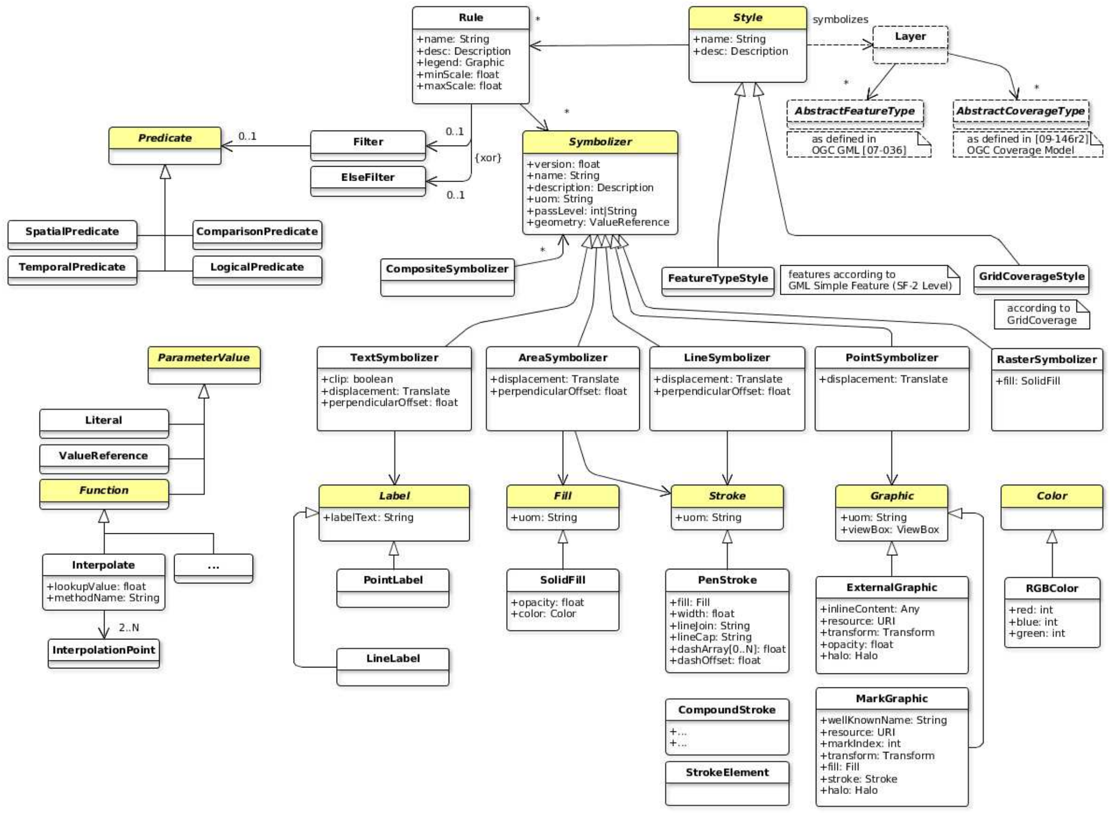
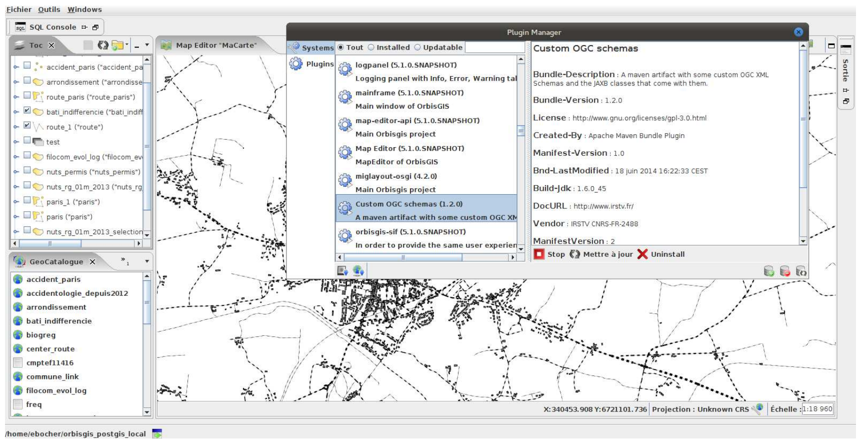


Figure 13. Extract of the proposed symbology model.


Fichier Outils Windows

SQL Console

Toc
- accident_paris ("accident_pa
- arrondissement ("arrondisse
- route_paris ("route_paris")
- [✓] bati_indifferencie ("bati_indiff
- [✓] route_1 ("route")
- test
- filocom_evol_log ("filocom_ev
- nuts_permis ("nuts_permis")
- nuts_rg_01m_2013 ("nuts_rg
- paris_1 ("paris")
- paris ("paris")
- nuts_rg_01m_2013_selection

GeoCatalogue
- accident_paris
- accidentologie_depuis2012
- arrondissement
- bati_indifferencie
- biogreg
- center_route
- cmptef11416
- commune_link
- filocom_evol_log
- freq

Map Editor "MaCarte"

Plugin Manager

Systems | Plugins | [✓ Tout] Installed Updatable

- logpanel (5.1.0.SNAPSHOT) Logging panel with Info, Error, Warning tal
- mainframe (5.1.0.SNAPSHOT) Main window of OrbisGIS
- map-editor-api (5.1.0.SNAPSHOT) Main Orbisgis project
- Map Editor (5.1.0.SNAPSHOT) MapEditor of OrbisGIS
- miglayout-osgi (4.2.0) Main Orbisgis project
- Custom OGC schemas (1.2.0) A maven artifact with some custom OGC XM
- orbisgis-sif (5.1.0.SNAPSHOT) In order to provide the same user experien

Custom OGC schemas

Bundle-Description : A maven artifact with some custom OGC XML Schemas and the JAXB classes that come with them.

Bundle-Version : 1.2.0

License : http://www.gnu.org/licenses/gpl-3.0.html

Created-By : Apache Maven Bundle Plugin

Manifest-Version : 1.0

Bnd-LastModified : 18 juin 2014 16:22:33 CEST

Build-Jdk : 1.6.0_45

DocURL : http://www.irstv.fr/

Vendor : IRSTV CNRS-FR-2488

ManifestVersion : 2

Stop | Mettre à jour | Uninstall

Sortie

X: 340453.908 Y:6721101.736 | Projection : Unknown CRS | Échelle : 1:18 960

/home/ebocher/orbisgis_postgis_local

Figure 14. OrbisGIS dynamic module system with OSGi.

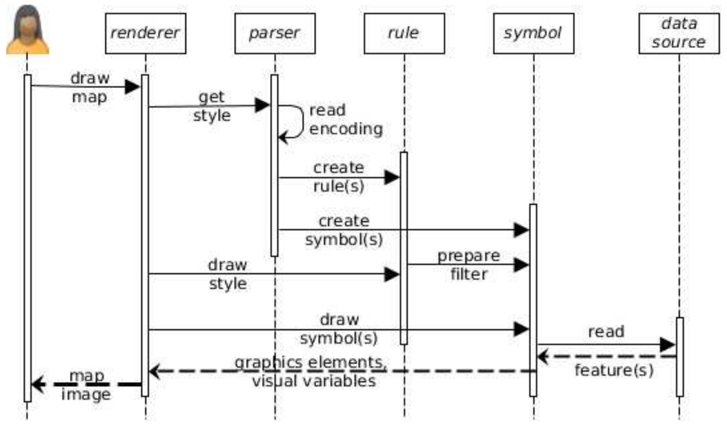
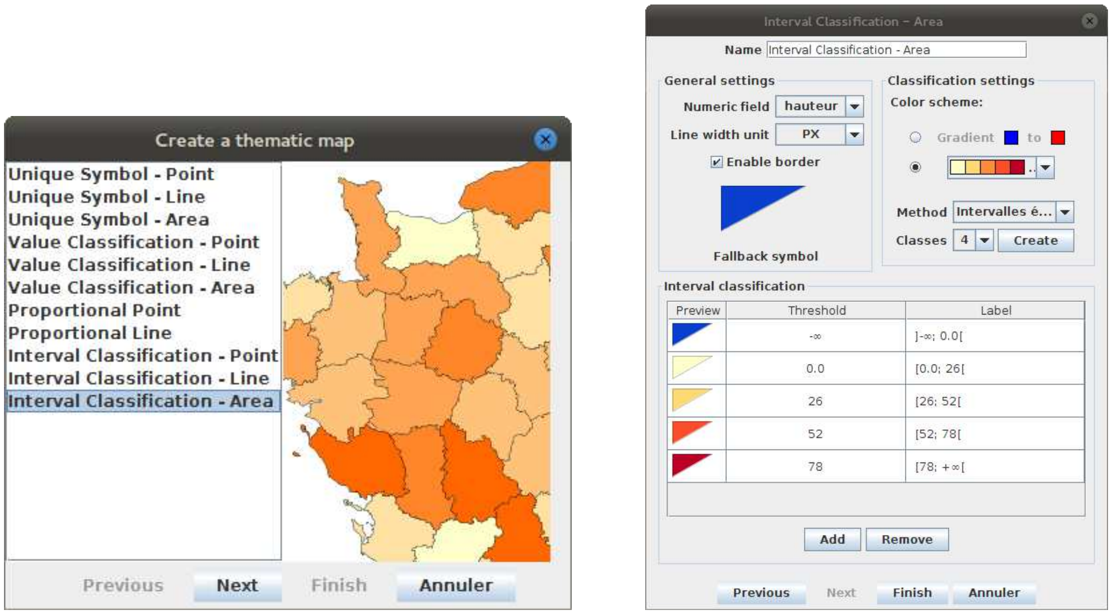


Figure 15. Main sequences of the rendering engine.


Create a thematic map

- Unique Symbol - Point
- Unique Symbol - Line
- Unique Symbol - Area
- Value Classification - Point
- Value Classification - Line
- Value Classification - Area
- Proportional Point
- Proportional Line
- Interval Classification - Point
- Interval Classification - Line
- Interval Classification - Area

Previous | Next | Finish | Annuler

Interval Classification - Area

Name: Interval Classification - Area

General settings

Numeric field: hauteur

Line width unit: PX

[✓] Enable border

Fallback symbol

Classification settings

Color scheme:

Gradient 🟦 to 🟥

Method: Intervalles é...

Classes: 4 | Create

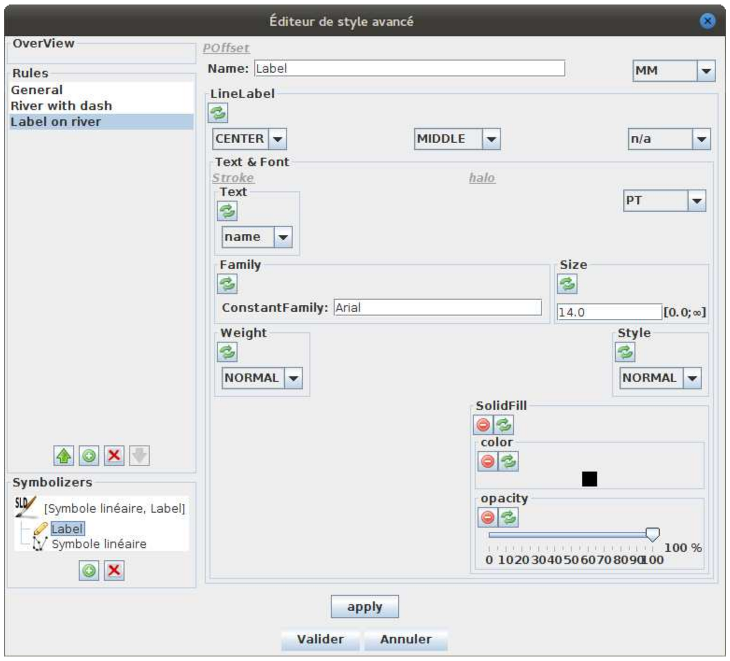
Interval classification

Preview	Threshold	Label
🟦	-∞	]-∞; 0.0[
🟨	0.0	[0.0; 26[
🟨	26	[26; 52[
🟥	52	[52; 78[
🟥	78	[78; +∞[

Add | Remove

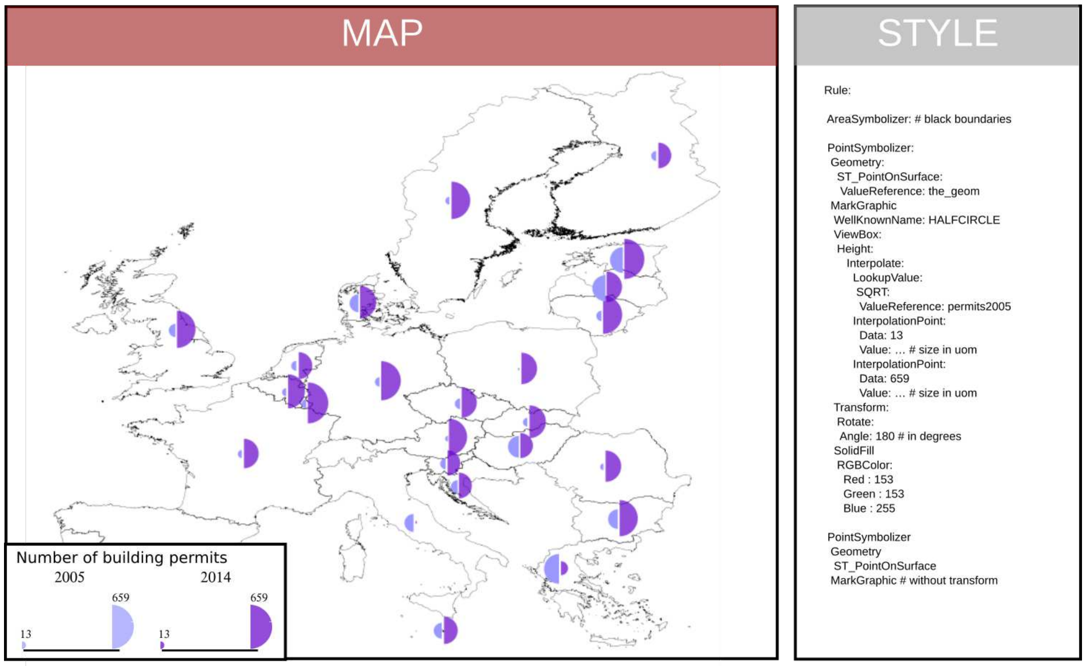
Previous | Next | Finish | Annuler

Figure 16. Screenshots of the user interface designed for productivity.



Name: Label
MM
CENTER
MIDDLE
n/a
PT
name
ConstantFamily: Arial
14.0 [0.0;∞]
NORMAL
NORMAL
100 %

Figure 17. Screenshot of the prototype of an advanced style editor.



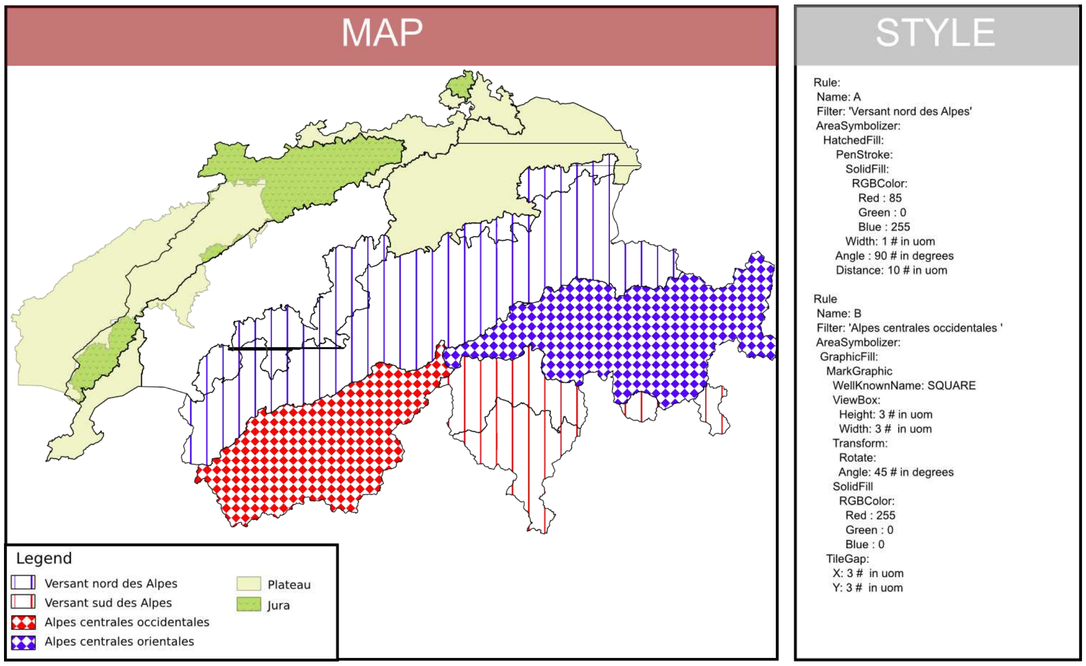
```
Rule:

 AreaSymbolizer: # black boundaries

 PointSymbolizer:
  Geometry:
   ST_PointOnSurface:
    ValueReference: the_geom
  MarkGraphic
   WellKnownName: HALFCIRCLE
   ViewBox:
    Height:
      Interpolate:
        LookupValue:
         SQRT:
          ValueReference: permits2005
        InterpolationPoint:
          Data: 13
          Value: ... # size in uom
        InterpolationPoint:
          Data: 659
          Value: ... # size in uom
   Transform:
    Rotate:
     Angle: 180 # in degrees
   SolidFill
    RGBColor:
      Red : 153
      Green : 153
      Blue : 255

 PointSymbolizer
  Geometry
   ST_PointOnSurface
  MarkGraphic # without transform
```

Figure 18. A bivariate proportional symbol map outcoming from the rendering of some redesigned symbology instructions (YAML-like encoded for the ease of reading).



```
Rule:
 Name: A
 Filter: 'Versant nord des Alpes'
 AreaSymbolizer:
   HatchedFill:
      PenStroke:
         SolidFill:
           RGBColor:
             Red : 85
             Green : 0
             Blue : 255
         Width: 1 # in uom
      Angle : 90 # in degrees
      Distance: 10 # in uom

Rule
 Name: B
 Filter: 'Alpes centrales occidentales '
 AreaSymbolizer:
  GraphicFill:
    MarkGraphic
      WellKnownName: SQUARE
      ViewBox:
        Height: 3 # in uom
        Width: 3 #  in uom
      Transform:
        Rotate:
        Angle: 45 # in degrees
      SolidFill
        RGBColor:
          Red : 255
          Green : 0
          Blue : 0
    TileGap:
      X: 3 #  in uom
      Y: 3 #  in uom
```

Figure 19. Combined visual variables to cartography the biogeographic regions in Switzerland.