SVG and the TOP10NL project

Barend Köbben <kobben@itc.nl>

Lecturer Cartographic Visualisation in the Department of Geo-information Processing,
International Institute for Geo-information Science and Earth Observation (ITC)
PO Box 6, 7500 AA Enschede, The Netherlands

Abstract

This paper presents the Scalable Vector Graphics (SVG) format as an important new possibility for cartographic visualisation. The main properties of the format are presented and emphasis is given to the role of SVG as part of the larger XML family and the possibilities this offers.

The second part of this paper is about the TOP10NL project, in which XML also plays an important role. TOP10NL, officially titled "the second generation TOPvector data-model project", was initiated by the Dutch National Mapping Agency, together with its user-community and the universities of Delft, Wageningen and the ITC in Enschede. The GML (Geography Markup Language - another member of the XML family), has been chosen for the data-exchange of TOP10NL. The first stage has been completed recently and to present the results a demonstration CD and website have been developed.

Web graphics today

Looking at the techniques that are used nowadays to disseminate maps, or graphics in general, on the World Wide Web, it becomes clear that the majority is stored in one of the standardised formats GIF and JPEG. The advantage is that these are truly standardised: they are accepted as the official World Wide Web Consortium's (W3C) standard graphics formats and every browser can display them natively. The most readily recognised disadvantage of these formats is their raster structure: the image is stored in pixels with a fixed resolution. This forces authors to either choose a low resolution image with inherent low graphic quality, or a higher resolution, with a better quality image, resulting in large file sizes and therefore long loading times.

A potentially even greater disadvantage is the fact that raster depictions are basically "stupid" visualisations of the data, in which the actual *information* is lost. Although one might clearly see a road, there is no such object present in the data and it is therefore difficult to work with the original data objects, for example for attaching interactivity or metadata to them. Text objects especially are reduced to an incoherent bunch of pixels, that cannot be searched or indexed. Because of that, an internet search engine will never find the toponym "Amsterdam" in a GIF-map of the Netherlands, a fact that might surprise the average user, because he can clearly read it on his screen…

Many existing websites might at first glance look like more sophisticated applications than described above. There are, for example, sites where a WebGIS system can be accessed on a server to toggle layers of information, zoom and pan, search by attributes or even to produce buffers and overlays. But most of these systems will eventually result in the server sending just a simple GIF or JPEG image for display in your browser window.

Besides the standardised raster formats, there are of course various other possibilities to disseminate graphics on the Web. These solutions can be used to overcome some or many of the disadvantages of standardised raster formats. This can be achieved by supporting vector graphics, object oriented structures, variable resolutions, interactivity and animations, among other things. But virtually all of these solutions use non-standardised formats, owned or patented by commercial companies (so-called 'proprietary formats'). One such format is the very popular Macromedia Flash format, and like most of these formats a separate browser plugin is required to use these files in your web browser. These plugins can usually be downloaded for free. But because the formats are binary and their specifications are not released publicly, one needs commercial 'authoring' software to produce these graphics.

Web graphics of the future

In order to overcome these disadvantages, one would prefer future Web formats that deliver good graphical quality in resolution-independent, small files. Furthermore, the actual information content should be preserved. The former demand can be met by choosing vector-based graphics. These would also accommodate the latter demand, but more is needed to meet that fully: There should be a structured relation between the format used to store the actual data and that used to visualise it. Preferably the format of choice should also be standardised by a non-commercial organisation, making the specification open and freely available, and the files easily creatable and editable. In light of these demands, it is not surprising that web developers looked into XML as a possible solution.

The eXtensible Markup Language (XML) has been designated by the W3C as 'the backbone of the future Web'. XML is based on SGML, the international standard metalanguage for text mark-up systems (ISO 8879). It is not a fixed format, but a 'metalanguage' —a language for describing other languages— that lets you design your own customized mark-up languages for limitless different types of documents. As such it is much more than a web page language, it's useable for storing and exchanging any kind of structured data (for more information, see eg. URL 3). Many XML-based formats have been developed, among them Geographic Markup Language (GML) , which is the OpenGIS standard for geographical information exchange.

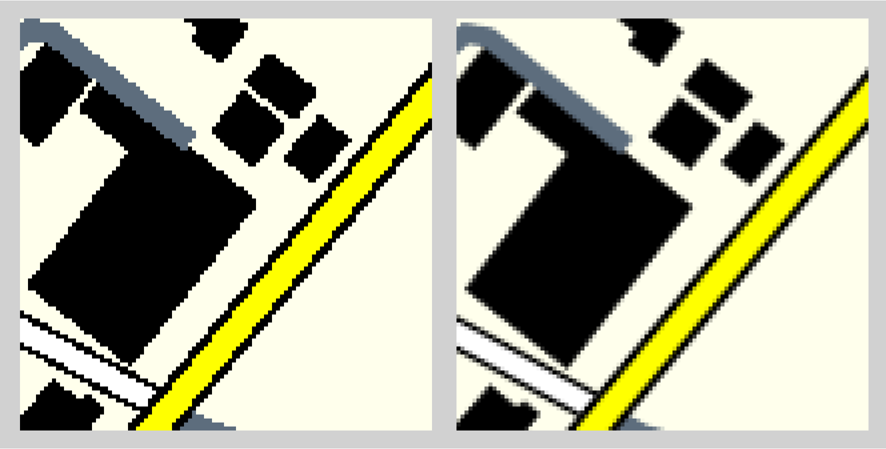
Scalable Vector Graphics (SVG)

Late 1998 a working group of the W3C has begun defining an XML-based graphical format: Scalable Vector Graphics (SVG). Already in 2000 the specification was almost stabilised and the first implementations became available and in September 2001 version 1.0 became an official standard or 'recommendation' in W3C jargon, recently followed by version 1.1 (defined in URL 1). In the next paragraphs, It will become make clear that SVG potentially is a very useful format, especially for cartography, because it offers graphic quality, scalability, interactivity and animation, as well as being fully integrated in the XML data flow.

SVG: graphic quality

The SVG standard offers virtually all known graphical primitives to build up vector elements and all usual operations of 2D vector graphics: paths that can be closed or open, their vertices can be connected by straight lines, arcs, ellipses, Bezier curves, and quadratic as well as cubic splines. Attributes of the elements control their line style, dash, joining, and miter. Paths can be can be filled with colour, gradients, patterns, etcetera. Probably the most obvious advantage of SVG is the quality of the visualisation. This is in large part due to the 'anti-aliasing' that is applied in the final rendering of the objects to the screen (see figure 1).

Figure 1. Comparison between normal (left) and 'anti-aliased' rendering (maps from URL 4, enlarged 400%).

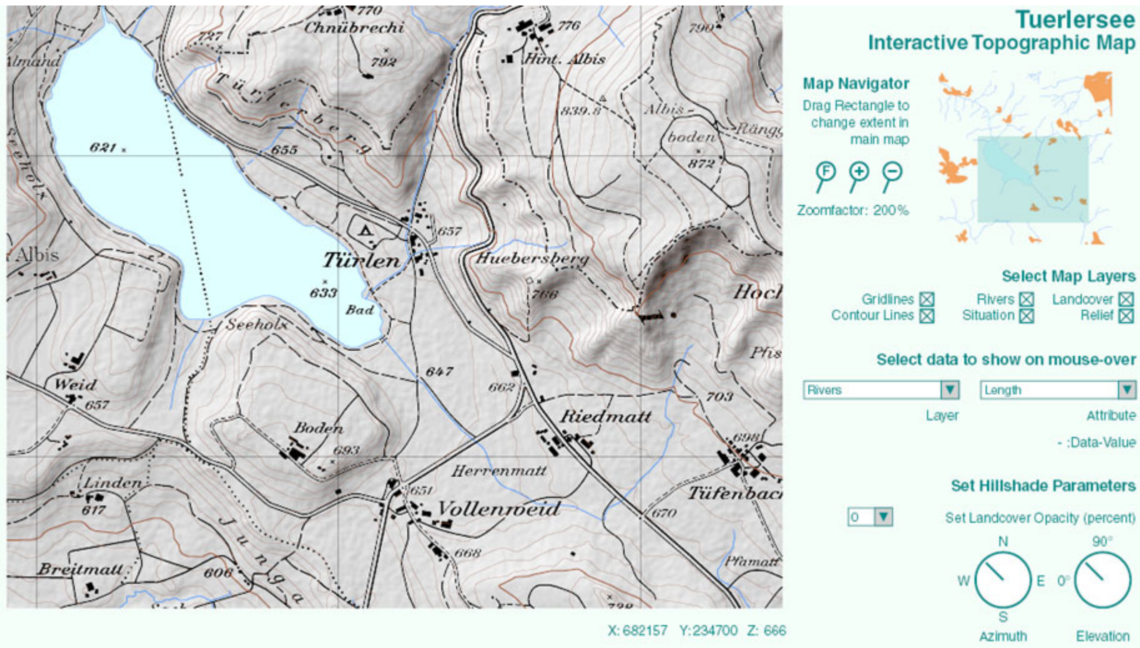


As well as vector graphics, raster images can be included, either in-line or referenced from external files, in JPEG, GIF or PNG format. Text is stored as text objects, thus the toponym "Amsterdam" is stored as a text string "Amsterdam" with attributes such as its location, the font (which can be stored internally or be a reference to a system font), its size, its colour, etcetera. Thus textual information can be searched and indexed and as the UNICODE system is used for glyph encoding, internationalisation is also taken care of.

All elements are available as objects in the so-called 'DOM-tree'. This 'Document-Object-Model' describes the whole graphic as a tree of inter-related hierarchically structured objects with a wide range of attributes such as colour, line width, transparency, etcetera. Al these attributes can be changed, either declaratively in the SVG format itself, or procedurally by scripting techniques.

A rather special addition to SVG is a set of filter primitives. These are applied to all or part of the graphics before final rendering on the screen. These filters, and especially their combinations, can be used to achieve all kinds of effect, from simple drop shadows to interactive hill shading (see figure 2).

Figure 2. Because of SVG's filter functions it is possible to have a hill shading in this map (based on a internally stored DEM) that can be interactively modified by the "dials" in the lower right of the user interface (source: URL 5).



SVG: scalability

Another strong point of SVG, especially for cartography, is its scalability. It is scalable because the vector elements are resolution-independent, and one can therefore zoom and pan, view on screen and print using all kinds of output hardware without loss of quality. But also the more traditional cartographic meaning of scaling is applicable, as all elements can have their own user defined 'coordinate-spaces', that can be transformed (rotated, skewed, translated, etc.) separately or grouped. Therefore, it is possible to combine within one SVG page several georeferenced coordinate systems. In the example in figure 2, the main map uses the Swiss national reference system, and the position of the mouse is shown using these coordinates in meters, plus the height derived from the DEM. At the same time the same coordinate space is shown at a smaller scale in the overview map, and these two systems are interconnected, because moving the small rectangle in the overview map will pan the main map window.

Scalability can also be referd to as the possibility to change the visualisation of the same set of SVG objects depending on user needs and hardware capabilities. This is done firstly by the use of 'Cascading Style Sheets' (CSS), another W3C standard. Using this technique, the same SVG file can be rendered in different ways, by applying different style sheets to it: for example one for viewing in colour in a web browser, another one for grey scale printing or a third one for a special case such as hydrographical charts to be viewed at night on the bridge of a ship. Secondly, the recently accepted version 1.1 of the SVG standard accommodates three different profiles, or subsets of the standard: *SVG full* for normal use on desktop machines and printers, *SVG basic* for more limited devices such as PDA's, embedded systems and the future UMTS phones, and *SVG tiny*, for devices with very little memory and tiny screens, such as WAP and GPRS phones.

SVG: interactivity

The DOM-tree mentioned earlier exposes all SVG elements and their attributes, making it possible to add interactivity to them. This can be done both in a declarative way, ie. interactivity described in the attributes in SVG objects themselves, or procedurally, using scripting in ECMAscript (the standardised JavaScript). An example of the former is the SVG snippet shown here:

```
<circle id="c1" cx="100" cy="100" r="50">
       <animate id="c1" attributeName="r" from="50" to="100" dur="3s"
       begin="c1.click"/>
</circle>
```

Here the "circle" object called "c1" has a child that is the "animate" object. This has a "begin" attribute which states that the animation should begin when an object called "c1" (the circle again) is being clicked. The same can be achieved using the procedural technique of scripting, as shown in this snippet:

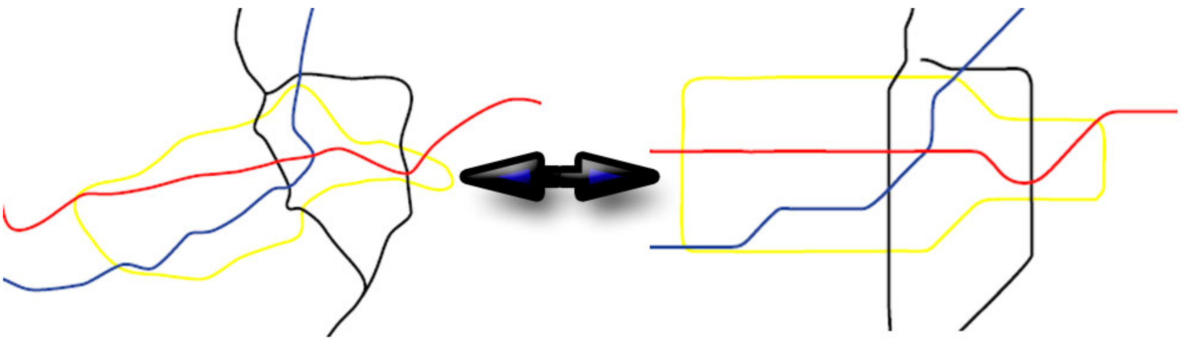
```
<script type="text/ecmascript">
       function clickCircle(evt) {
       svgdoc.getElementById(c1).setProperty("begin","true");
       }
</script>
```

These techniques can be used to effectively build up complex interactive mapping applications, such as those in URLs 4, 5 and 6. The map of Vienna in URL 6, constructed in 2000 by Andreas Neumann at the ETH in Zürich, was one of the first showcases of the potential of SVG and served as an important 'eye-opener' for many cartographers.

SVG: animated & dynamic maps

The SVG standard supports the use of declarative animation by incorporating yet another W3C standard for animation called SMIL. This results in the possibility of dynamically changing virtually all attributes of objects using time-based parameters. The first example snippet in the previous paragraph describes the change of the radius of the circle object from 50 to 100 units over a period of 3 seconds. Because of the time-based nature of the parameters, it is simple to have various attributes dynamically changing independent of each other, something which is much harder to achieve using the more traditional frame-based animations (such as used in Macromedia's Flash). Thus animations such as the one shown in figure 3 (and URL 6) can be achieved using simple coding and only a small increase of file size.

Figure 3. Animation of the location attributes (x,y) of SVG-paths: Part of the London Underground as a 'morph' from the geographic (left) to the schematic depiction (right) and back (see URL 6).



SVG Implementation

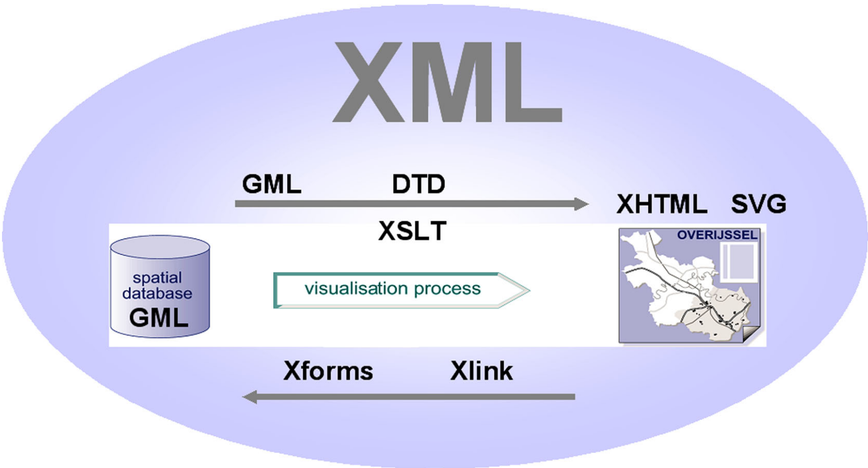
Since the SVG specification was accepted in September 2001, both commercial and non-profit organisation have started implementing it. Of course, with it being a W3C standard, it is expected that in time all major web browsers will support it natively. But at the moment, one still needs plugins to make SVG graphics work. Adobe released 'SVGviewer', now in version 3 and running on major browser on both MacOS and Windows. Corel has recently added an SVG plugin as part of their upcoming SmartGraphics platform, and also some standalone players are available (Batik on Java, kSVG for Linux, the Mozilla project, etcetera). Another development is implementation for PDA's and so-called 'embedded systems', where SVG is expected to become an important format. All in all, there is a growing and largely stable support for viewers.

For the authoring of SVG the possibilities are more limited. There are a number of existing drawing and GIS softwares that can export to SVG (among them OpenOffice.org, Adobe Illustrator, the MAPublisher plugin for Macromedia's Freehand and the SVGMaker extension to ArcView). There are also solutions to generate SVG server-side (eg. using perl or php), and stand-alone converters (such as PS-SVG and SVG printer drivers). All these solutions are 'up and running', but limited in their possibilities, especially when one wants to use the features which are special to SVG, such as the filters, animation and interactivity. Stand-alone SVG authoring software which does support these features is becoming available: Jasc WebDraw is now in version 1.03 and Evol-Grafix (formerly PCX) has just released Xstudio. Both of these can be said to be not yet matured, and for cartographers especially lacking links with existing mapping/GIS software.

SVG = XML (= GML = XHTML = Xforms = …)

Arguably the most important strength of SVG is the fact that is an XML language and as such fits seamlessly in the world of XML based formats. This makes it possible to realise cartographical applications as part of a workflow based on an open, distributed and component based Web environment, such as outlined in figure 4. An example of such an environment could be theTOP10NL project, which will be described in the next paragraphs.

Figure 4. Possible set-up of the cartographic workflow within an XML environment.



The TOP10NL project

TOP10vector, based upon maps scaled 1:10.000, has been digitally available since 1997 for the complete coverage of the Netherlands. TOP10Vector is basically a digital map, not a topographic database. It has been structured on the basis of a coding system used for cartographic reasons, and as such has become a mixture of a landscape model and a cartographic model. The feedback of users has made it clear that there is a need to redesign the structure, contents and dissemination if the data is to become more useful in GIS applications.

Together with the users, the universities of Delft, Wageningen and the ITC in Enschede, the Dutch National Mapping Agency (Topografische Dienst) has started the TOP10NL project, officially titled *"the second generation TOPvector data-model project"*. The first stage, which has been completed recently, comprised the determination of the user requirements, the design of a new data-model, the production of prototypes and the evaluation of these prototypes in relation with the formulated requirements. To present these results to a larger group of users, potential users and others who are interested, a demonstration CD and website (URL 12) have been developed.. Visitors of the site can evaluate the data-model, examine the prototypes and are invited to give comments.

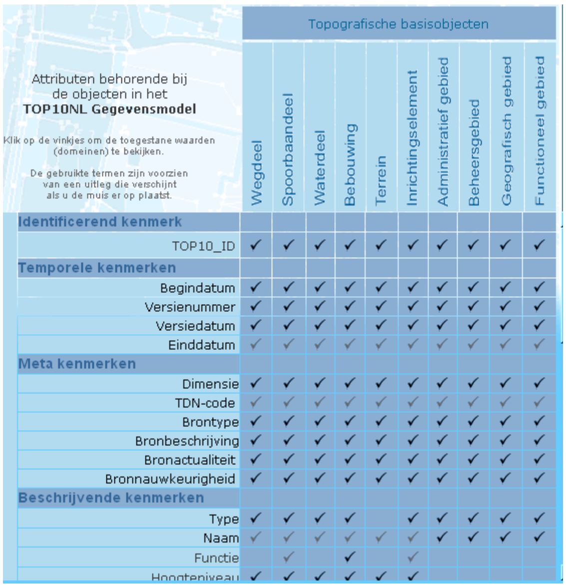
The newly designed data-model and the produced prototypes are only the beginning of the process, which will result in a totally new designed database for all TOPvector products. Some of the other projects are the design of a new Digital Cartographic Model for visualisation purposes, the design of the smaller scale data-models and research into the possibilities of incorporating 3-D information in TOP10NL. It is expected that the new products will be available ‘on the market’ in 2005.

A new data structure

Requirements for new data structure have been formulated by the National GI platform. The new structure has to be object oriented, using unique ID's and should be stored in a seamless databases with change-only data. The data objects should support multiple representations, eg. roads should be represented by their physical area (as polygons) as well as by their network structure (as nodes and lines). This would also serve to support automatic generalisation for smaller scales as much as possible. Because linkages to other datasets are essential, compliance with OpenGIS standards was also demanded.

The new data-model for TOP10NL contains a set of *topographic basic objects*, related to the visualisation on a scale 1:10.000, which are defined as entities in the data-model. These topographic basic objects have identifying, temporal, meta and descriptive characteristics which are the attributes in the data-model. Figure 5 shows part of the matrix that describes the relations between basic objects, their attributes and their constraints.

Figure 5: part of the basic objects matrix of TOP10NL (in Dutch: identificerend kenmerk = identifying characteristic, temporele = temporal, beschrijvende = descriptive)



Topografische basisobjecten

Attributen behorende bij de objecten in het **TOP10NL Gegevensmodel**

Klik op de vinkjes om de toegestane waarden (domeinen) te bekijken.

De gebruikte termen zijn voorzien van een uitleg die verschijnt als u de muis er op plaatst.

	Wegdeel	Spoorbaandeel	Waterdeel	Bebouwing	Terrein	Inrichtingselement	Administratief gebied	Beheersgebied	Geografisch gebied	Functioneel gebied
Identificerend kenmerk										
TOP10_ID	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Temporele kenmerken										
Begindatum	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Versienummer	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Versiedatum	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Einddatum	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Meta kenmerken										
Dimensie	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
TDN-code	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Brontype	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Bronbeschrijving	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Bronactualiteit	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Bronnauwkeurigheid	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Beschrijvende kenmerken										
Type	✓	✓	✓	✓		✓	✓	✓	✓	✓
Naam	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Functie		✓		✓		✓				
Hoogteniveau	✓	✓	✓	✓	✓	✓				

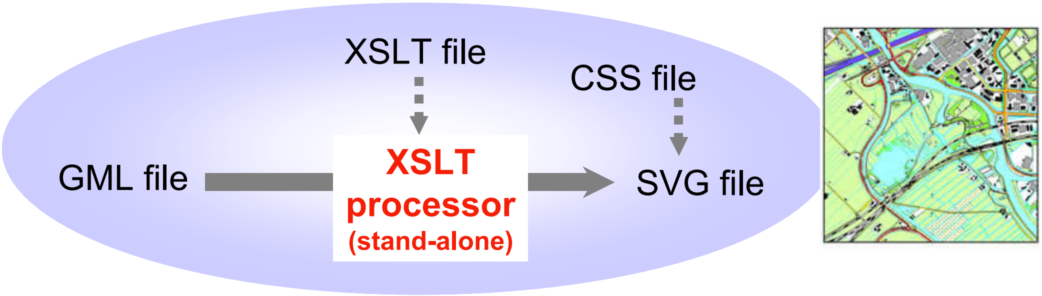
The topographic basic objects are the basic elements for building *complex objects*. For example a road is a composite of road segments, which are known under a certain name, e.g. "A1" or "Hengelosestraat". Complex objects are not available as objects in de database, but can be created by linking basic objects with common characteristics (e.g. a road-number or a street-name). For generalisation purposes complex elements can be derived by common characteristics. For example a ‘forest’ object in smaller scales is a collection of terrain objects which is characterised by the land-uses ‘deciduous forest’, ‘coniferous forest’ and ‘mixed forest’.

GML and SVG in TOP10NL

Because of the requirement of confirmation to OpenGIS standards, the logical choice for the exchange of TOP10NL data was the Geography Mark-up Language. GML, another XML language, is being developed within the OpenGIS Consortium as their standard for geographical information exchange (see URL 2). The GML specifications consists of three XML Schema documents (geometry.xsd, feature.xsd and xlinks.xsd) who together define the framework on which users can base their own GML format. Such a user-specific GML Schema has been designed for the TOP10 project on basis of the new data-model. In the TOP10NL Schema a number of object-types are defined: Terrain, Road segment, Water segment, Railroad segment, Building, Specific terrain elements (eg. utility), Administrative boundaries, Geographic area, Functional area, Conservation area, etcetera. The object-types belong to the Feature Collections and there is one basic or root collection, named TOP10 Themes. GML specifies only that there must be a structure for Feature Collections, Feature Members and Features, not how these groups of information specifically are defined. In the present prototype for example terrain, road segment, water segment and railroad segment are grouped under the collection Spatial Objects, with the reason that these four object-types in principle do not overlap each others (except when crossing bridges, tunnels or viaducts) and in this way are closely related. Other object-types are not merged to one collection, but if after the evaluation users prefer a different grouping it can be easily realised within GML. GML can be used directly to visualise the data from a web server, such as has been tested by the company Ionic software, that has put TOP10NL test data in their OpenGIS Web Feature Server (see URL 13).

Using SVG for visualisation of TOP10NL is again logical, because SVG is also XML and therefore combines naturally with GML. Furthermore it was demonstrated in the first part of this paper that SVG is a cartographically sound format. Test have been done to experiment with this in several workflows. One test involved using “SVG Maker” to convert shape files to a static SVG colour map according to the current Top10 output specifications. More interesting is the work done by the Technical University of Delft, using XSLT transformations to directly process GML files into SVG (shown in figure 6).

Figure 6: workflow for SVG visualisation of GML data (see URL 14).



But more possibilities have to be explored, for example the incorporation of GML objects directly in databases and the possibilities of "on the fly" processing, where GML data objects are converted on-line by a server application and 'pushed' to the clients SVG browser. Also there are some problems to be solved in both the TOP10NL data model and in its expression in SVG objects. One such problem is the correct visualisation of overlapping objects, such as roads crossing a river. Furthermore some attention should be given to the optimisation of generated SVG, in terms of file size as well as rendering speed.

URL's

1. W3C SVG Recommendation: http://www.w3.org/TR/SVG/
2. Galdos systems – GML technology: http://www.galdos.ca/technology-index.html
3. The XML FAQ: http://www.ucc.ie/xml/faq.xml
4. Webmapping and Cartographic Generalization Prototype 0.3a: http://www.geo.unizh.ch/gis/research/webmap/proto/proto.html
5. Tuerlersee - Interactive Topographic Map Example: http://www.carto.net/papers/svg/tuerlersee/
6. Vienna - social patterns and structures: http://www.carto.net/andi.n/about_vienna_svg.html
7. London Underground morph (geographic vs. schematic): http://kartoweb.itc.nl/public_examples/UndergroundMorph.svg
8. Proceedings of SVG Open 2002: http://www.svgopen.org/index-2002.shtml
9. carto:net - cartographers on the net - scalable vector graphics: http://www.carto.net/papers/svg/index.html
10. SVG café: http://www.svg-cafe.com/
11. Adobe SVG zone: http://www.adobe.com/svg/
12. TOP10NL site (in English): http://kartoweb/top10nl/TOP10NL_eng/
13. Ionic software's GML in an OpenGIS Web Feature Server (in English): http://demo.ionicsoft.com/demos/tdn/
14. SVG visualisation of TOP10NL data using XSLT transformation (in Dutch): http://webmap.geo.tudelft.nl/proto/

NB: for viewing most of these sites an SVG-plugin is necessary. Currently the best supported and most used of these is the Adobe SVG-viewer 3.0 (see URL 11).

Literature

If you want to find out more about SVG, URLs 9 –11 are arguably the best starting point. Furthermore a number of books has been published about SVG and its applications; one of the most extensive and detailed being:

*Watt, Andrew et al. (2003): **SVG unleashed.** SAMS publishing, Indianapolis. ISBN 0-67232-429-6.*