XML TECHNOLOGIES IN WEB BASED GEOGRAPHIC INFORMATION SYSTEMS

B. Predić, D. Stojanović
Faculty of Electronic Engineering, University of Niš
Beogradska 14, 18000 Niš, Serbia and Montenegro
{bpredic, dragans}@elfak.ni.ac.yu

Abstract: In this paper we present possibilities of using XML based data formats in future Web geographic information systems (GIS). We mainly focus on XML based vector graphics format SVG and its impact on performance of ultra thin web clients for GIS. A new application architecture is needed in order to take full advantage of mentioned vector graphics format. The proposed architecture of such a system implementing this technology is developed and described. Vector graphics (SVG) is used to present geographic data to users. There are also XML based data formats for storing and exchange of geographic data. In this paper we briefly show GML, Open GIS Consortium defined XML based data format especially tailored for geographic data.

Keywords: XML, Internet, WEB GIS, vector graphics, W3C, GML, SVG

1 Introduction

With the explosive growth of the Internet, migration of geographic information systems (GIS) into the WEB was natural and expected. This migration caused the need for standardization in this field.

Early Web GIS solutions exploited technologies already available to Web designers and programmers. Unfortunately, HTTP protocol was designed to transfer mainly textual data. Graphics can be added to Web presentations but has to be limited in size not to overstrain Internet bandwidth.

In the case of Web GIS, focus of the application is the map that is usually an image. It has to be large enough to contain enough information important to the user. The most common approach to date was to present the map in the form of raster image. It perfectly fits the concept of HTTP which is request/response based. The user (client) forms the request and transmits it to the server. At the server data is loaded or dynamically created and transferred back to the client where it is being displayed. This is known as a round trip.

However, there are a couple of drawbacks to this architecture. First, and the most obvious one to the users, is long response time. Raster images in resolutions high enough to be used in GIS applications as maps can be very large. It takes considerable amount of time to transfer that volume of data over the Internet.
Another limitation is only basic interactivity raster images offer. Raster images are only sets of pixels without any logical structure regarding geographic objects they represent. It is difficult to distinguish which object user is interacting with. Also, every user action initiates a new round trip with appropriate response delay.

Aim of our research is to explore possibilities vector graphics offer to Web GIS developers, especially focusing on Scalable Vector Graphics (SVG), emerging standard in this field defined by World Wide Web Consortium (W3C). Expected results are performance increase, shortened response times and increased interactivity of the systems. GML contributes to standardization of geospatial data storage and especially exchange formats allowing easy integration of different GIS applications and components distributed over Internet.

2 Web based GIS

An online GIS would allow a user on one computer to access geospatial data that resides on one or more remote computers using Internet technologies.

Since Web based GIS applications are limited by available bandwidth of the Internet, architectures different to those used in desktop GIS applications have to be implemented. Another limiting factor for Web GIS developers is heterogeneity of hardware used to access WEB based GIS applications. This heterogeneity is not only reflected in hardware, but more importantly in software clients use (operating system and Internet browsers).

So far, the most common approach was to create the thinnest client as possible capable of displaying maps in the form of bitmapped image (Fig. 1.). This maximizes the availability of created Web GIS application on heterogeneous systems. All computing intensive work is done on the server. With the growing number of potential users demands for server computing power are increased dramatically. More importantly this approach limits interactivity of client applications. It also causes inefficient usage of already limited network bandwidth since every user interaction initiates creation of raster map containing information requested by user and data transfer from the server to the client. This process occurs even if considerable amount of data user requested already exists on the client. Java applets or other forms of Internet browser plug-in applications add extended interactivity to the client side at the cost of possibly lengthy download before user can even start working with Web GIS. Either solution produces slow response time of the system, something casual users on the Internet today do not tolerate. Also, there is always the possibility that after this lengthy process information user requested is not available at all.

There are a number of projects dealing with Web GIS already available. These are proprietary solutions developed independently by leading companies in this field: AutoDesk’s MapGuide, ESRI’s ArcView IMS, Intergraph’s Geomedia WebMap and others.

Recent technical developments by the World Wide Web Consortium (W3C) provide a foundation for non-proprietary, open source Internet Mapping [4]. W3C has been ac-
tive in extending the Internet through a new generation of standards based on XML. These standards allow for distributed, interactive, scalable Internet GIS applications. Interactivity is the key distinguishing feature between paper and computer maps and it is imperative to retain this characteristic in Web GIS applications with the minimal degradation of performance.

![Diagram of architecture performing all transformations on the server]

**Fig. 1: Architecture performing all transformations on the server**

### 3 XML technologies in Web GIS

The basic idea behind XML is not new. It is in fact a simplified version of Standard Generalized Markup Language (SGML). HTML is based on XML and its every modification, future or underway at W3C, is based on XML. In order to make future Web GIS applications easily deployable on the Web it is preferable to base all formats for storing and visually presenting geospatial data on XML. Two XML grammars are of special interest to Web GIS developers: Geography Markup Language (GML) and Scalable Vector Graphics (SVG).

GML is defined by OpenGIS Consortium (OGC) and used to store information about geographic entities [5]. It is important to draw some clear distinctions between geograhic data which is stored in GML and visual representation of that data that might appear on the map. Just as XML is helping the Web to clearly separate content from presentation, GML is doing the same in the world of GIS.

GML is based on the abstract model of geography developed by OGC. This model defines all real world geographic objects as entities called *features* (Fig. 2.). The state of each feature is defined by a list of properties with geometry as the most important one. Each property has the name, type and value. Geometry of a feature consists of
basic geometric entities: points, lines, curves, polygons and homogenous and heterogeneous collections of them (Fig. 3.). GML specification is for now restricted to two dimensional space.

Aggregate geographic object are encoded in GML as collection of features. There is no limit to the level of aggregation. Similarly, geometry of a geographic feature can be composed of many simple geometry elements. A real world example of feature aggregation would be a factory complex. It is a geographic feature with a number of properties (number of employees, etc.) and geometric characteristics (bounding box, outer fence, etc.). This complex feature consists of multiple separate features (buildings) described by their own properties (purpose, number, number of employees, etc.) and geometry (area, bounding box, etc.).

Structure of XML documents is defined by corresponding schema document. GML document must conform to three schemas: feature, geometry and Xlinks. The feature schema defines basic characteristics of a geographic feature such as feature ID (fid), name, description. The geometry schema defines geometric characteristics that can be associated with a feature. Possibility of creating hyperlinks between features contained in one GML document or even in multiple documents distributed over the network is defined by Xlinks schema. This is one of the most interesting and powerful characteristics of GML enabling easy integration of data collected by different users and stored in different locations. User defined schemas extend and customize these three basic schemas to best fit specific needs of the user.

Following is an example taken from the GML data used in our sample application “Niš City Guide”. It contains data for one of the streets.

To summarize, GML was developed to:

- Separate storage from visual presentation of geographic data
- Define standard for exchange of geographic data
- Provide means for integration of geographic data with other XML encoded data
- Enable creation of complex and distributed geographic data sets by implementing XLink

Storing and exchanging geographic data is only one half of the problem. The second half, and the one most visible to the end users, is visual representation of geographic data. To successfully present data to the user we should choose a graphics format that is scalable, easily created from GML data, easily integrated with HTML and the one that provides the greatest level of interactivity to the user. Graphics format that conforms best to these demands is Scalable Vector Graphics (SVG).

SVG is an XML grammar used for describing two-dimensional graphics [6]. It’s Data Type Definition (DTD), publicly specified by W3C, defines tag elements for shapes, images, text, and animation. This tag language fully conforms to XML, which allows use of standard XML tools such as validating parsers, editors and browsers. SVG graphics is stored as textual documents. This makes it perfect fit for exchange over the Web which was built on text exchange. SVG files contain large percentage of descriptive tags in comparison to raw data that is of interest to Web GIS users. Some
may question efficiency of transferring such textual files. Files containing SVG graphics can be GZipped prior to transfer over the Internet so there is no question of inefficient Internet bandwidth usage. Following is a simple example of uncompressed, plain text, SVG graphics file:

```xml
<cityMember>
  <Street fid="ST2">
    <gml:description>Seče pešačku zonu Pobedine ulice.</gml:description>
    <gml:name>Nikole Pasica</gml:name>
    <linearGeometry>
      <gml:LineString>
        <gml:coordinates>7572755.76,4797532.63 7572866.25,4797519.93 7572882.76,4797517.39 7572946.26,4797504.69 7573017.38,4797494.53 7573155.81,4797466.59 7573225.66,4797453.89 7573315.83,4797474.21 7573378.06,4797490.72 7573411.08,4797470.4 7573430.13,4797414.52</gml:coordinates>
      </gml:LineString>
    </linearGeometry>
    <type>ulica</type>
  </Street>
</cityMember>
```

Fig. 2: UML representation of feature schema
SVG is developed by a group of industry leading companies: Adobe, Macromedia, Sun, IBM, Microsoft and others. At the moment Internet browser needs a plug-in installed to be capable of rendering SVG graphics [1]. Since it is a standard supported by W3C it is reasonable to expect it to be integrated in future versions of Internet browsers.

Since it is a proper XML document SVG data can be accessed using Document Object Model (DOM). SVG DOM allows access to all elements, attributes and proper-
SVG offers a rich set of event handlers such as `onmouseover` and `onclick` which can be assigned to any SVG graphics element. This characteristic of SVG offers a new level of interactivity that was impossible for Web GIS applications using raster image maps. SVG DOM can be accessed using any scripting language such as JavaScript in the same manner as scripting languages are manipulating parent HTML document DOM. In fact, SVG DOM and HTML DOM integrate so seamlessly that scripting procedures attached to HTML document can access SVG data and vice versa without noticeable change in syntax when crossing boundary between text and vector graphics.

Once a SVG graphics is loaded and displayed in browser it can be panned, zoomed or modified by attached scripting procedures without the need for contacting server or refreshing the page [2]. These repeated *round trips* were the main disadvantage of using raster images for creating Web GIS. Another huge advantage of SVG is the possibility of incremental download of data. In a real life situation it means that graphics elements are displayed as they are being downloaded. Graphics elements can be grouped in such a manner that user gets basic shapes of the map almost instantly and gradually more details are drawn as map is being downloaded. It is now much simpler to implement different levels of detail on one map depending on zoom factor. As the user is navigating the map (zooming and panning) data is dynamically downloaded from the server in the background and appended to the SVG document. Again, there is no need for visually unappealing refresh of the whole web page. Data that is once downloaded and appended to the SVG document is memorised during the whole session and all panning or zooming back requires absolutely no data transfer from the server.

Another characteristic that is of interest to Web GIS developers is grouping of geometric objects into layers that can be dynamically turned on or off. Programmers can easily implement decluttering of maps this way. User can customize a map according to his needs simply by turning on or off different layers containing separate classes of geographic objects.

Captions and labels present on the map are stored as plain text and can be indexed and searched. This feature is present “out of the box” in SVG and it doesn’t take much coding to implement a fully text searchable map.

To summarize all advantages SVG offers to GIS developers over raster image formats [3]:

- It is stored in plain text format so it can be edited by a range of fairly simple tools and interpreted by humans. It is also highly compressible as all text files are.
- It is scalable. Since it is vector graphics format shapes are rendered using mathematical formulas and there is no image quality degradation at high zoom factors.
- All text contained in SVG document can be searched and selected and even indexed by Internet search engines.
• It is dynamic and highly interactive. Graphics objects can be individually selected and manipulated. All interaction is confined within the browser, there is no need for network round trip to the server and back.

• All objects can be animated either using SMIL and separate animation elements incorporated in SVG or JavaScript in conjunction with DOM.

• It is an open source solution. SVG is an open recommendation developed by a cross-industry consortium. It can be applied to Web GIS projects using entirely open source tools thus reducing the cost of implementation.

Both standards used for storing (GML) and visual presentation (SVG) of geographic data share common background, XML. So, it should not be difficult to perform transformation from GML into SVG. It would be logical to use a tool already available in the XML family. We found that eXtensible Style sheet Language for Transformations (XSLT) is up to this task. XSLT is declarative programming language. With XSLT programmer describes for pre-processor what to do, and not how to do it. This is the method programmers today are usually not used to, but it can be very effective. Programmer can focus on the problem and not on the auxiliary algorithms, variables and structures that are only helping to solve the problem. XSLT is based on templates. Templates describe what information pre-processor should look for in the source document and with what structures it should be replaced in the destination document. Since XSLT is not the focus of this paper we will not go into any more details here [7].

4 Niš City Guide

Prototype application we are discussing here is developed as part of the project “Geographic Information System for Local Authorities based on Internet/WWW Technologies”, funded by Ministry of Science, Technology and Development, Republic of Serbia and Niš Municipality. The purpose of this prototype was to evaluate usability of vector graphics in Web based GIS applications.

The architecture adopted for this prototype is a hybrid between ultra-thin and thick client architectures. Data extraction from the geospatial database, GML creation and transformation of GML into SVG is performed on the server. Produced SVG data is then transferred over the Internet and appended to the SVG document at the client. Client application handles all interactivity functionality.

Application is conceived as online tourist guide (Fig. 5.). It contains information that can be of interest to tourists visiting city of Niš and help them navigate around the city. Server side of the application runs on Microsoft Internet Information Server (IIS). Server side scripting is performed using ASP technology and VBScript language. Server also handles XML (GML and SVG) validation and parsing using Microsoft MSXML tools. Application is served geographic input data in GML format. That data is then being transformed into SVG used for visual presentation. Performed transformations include coordinate transformation from some geographic reference system into SVG screen coordinates. Not all geographic objects stored in geospatial
database are sent to the user. Based on the user defined viewport a sort of “clipping” is performed and only those objects visible in the viewport are encoded into the GML.

Fig. 4: Architecture implemented in “Niš City Guide”

The client side of the application consists of Adobe’s SVGViewer and JavaScript functions enabling interactivity. These are running within standard Internet browser like Internet Explorer or Netscape Navigator. Apart from basic functions like zooming and panning that are functionalities of SVGViewer users can retrieve name of the object cursor is pointing at, select specific object by name from drop-down lists formed for each class of objects presented or follow hyperlink to the page containing additional information about selected object by clicking on it on the map. Another standard GIS functionality implemented is grouping of objects of the same class into layers that can be separately shown or hidden.

At the moment application is displaying both vectorized data and raster background images. Raster maps are in separate layer and can be turned off. Also, raster map is divided into segments that are acquired from Web Map Server over Internet and cached locally. Caching significantly improves application response time on zooming and panning. This feature demonstrates interoperability between distributed GIS applications. Example application is available at: http://gislab.elfak.ni.ac.yu/bpredic/gmlnisrc/karta.asp.

5 Conclusion and Future Work

Analyzed XML grammars GML and SVG are introducing a whole new set of characteristics into the world of Web GIS. They are clearly separating content from the presentation of geospatial data. This was the cornerstone in Web development industry in general, allowing much cleaner designs of Web presentations and applications.
Since both GML and SVG have roots in XML the transformation between these two formats using another XML grammar, XSLT, is natural and easy. Maps are by nature vector layered representations of 2D space. SVG is based on these very concepts thus making mapping ideal application of SVG. Further, SVG localizes much of the interactivity on the client side eliminating the need for constant refreshes of the Web page and finally making response time of the Web application much shorter. Finally, less waiting on the WWW is something casual Internet users always appreciate.

Fig. 5: User interface of the application as seen in Internet Explorer

Future work will most probably include implementation of previously described technology in some sort of location based services and experimenting with the possibility of porting created application to handheld and mobile devices using SVG Basic and SVG Tiny.

6 References


