Making Maps With

Geography Markup Language (GML)

Galdos Systems Inc.

October 25, 2000

Contact: Ron Lake
Phone: (604) 484-2750
E-mail: rlake@galdosinc.com
1 Introduction

This paper is the second in our series on Geography Markup Language (GML), a new way to deal with geographic information via the Internet. This paper shows you how to make maps from GML. It introduces key ideas such as Map Styles and shows that with your browser, and not much else, you can make some pretty cool maps from GML data.

GML, like many Internet technologies, maintains a strong separation between presentation and content. You may remember from the first article that GML is concerned with the description of geographic content. It describes the properties of geographic entities (features) or distribution functions (coverages) using XML. Other tools and technologies, all of them based on open standards, are used to style GML into graphical presentations or maps.

2 What is GML?

2.1 Status

GML is an XML-based encoding standard for geographic information developed by the OpenGIS Consortium (OGC). Its current status is as a Recommendation of the OGC that was passed in May, 2000. This recommendation resulted from the efforts of several companies including Galdos Systems Inc., Laser-Scan Inc., Oracle Corporation, CubeWerx, Compsult Ltd and the U.S. Census Bureau. GML was originally implemented and tested through a series of demonstrations that formed part of the OGC’s Web Mapping Test Bed (WMT) conducted in the summer of 1999. These tests involved GML mapping clients interacting with GML data servers and service providers. Further exploration and development of GML is now taking place within the OGC’s Web Mapping Testbed Phase II and involves companies such as Galdos Systems Inc, CubeWerx Ltd., ESRI Ltd, DTIA, Ionic Software, and Laser-Scan Ltd. This includes in particular an XML Schema implementation of GML along with extensions for temporal features/events, feature relationships (geolinks), and coverages.

2.2 Geography, Graphics and Maps

Before we look at GML itself, it is important that we draw some clear distinctions between geographic data (which is encoded in GML) and graphic interpretations of that data as might appear on a map or other form of visualization. Geographic data is concerned with a representation of the world in spatial terms that is independent of any particular visualization of that data. When we talk about geographic data, we are trying to capture information about the properties and geometry of the objects that populate the world about us. How we symbolize these on a map, the colors or line weights we choose to use is something quite different. Just as XML now helps to clearly separate content
from presentation for web pages, so GML will do the same in the world of geography and mapping.

GML is concerned with the representation of geographic data content. As well, we shall see we can also use GML to make maps. This might be accomplished by developing a rendering tool to interpret GML data. However, this would conflict with the GML approach to standardization, and to the separation of content and presentation. To make a map from GML we need only to style the GML elements into a form that can be interpreted for graphical display in a web browser. Potential graphical display formats include W3C Scalable Vector Graphics (SVG), Vector Markup Language (VML), and the Web 3D Consortium’s X3D.

3 Making a Map with GML – An Overview

To make a map with GML data, you must style the GML geographical content into a suitable graphical presentation. Typically this involves the interpretation of the GML content using graphical symbols, line styles and area or volume fills, and often some sort of transformation of the geometry of the GML data into the geometry of the visual presentation. We refer to this interpretation process as map styling. This loosely corresponds to the more conventional document styling of XML into a presentation format such as HTML. In fact, many of the same tools and technologies, such as XSLT (XML Transformation Language), can be used to perform this styling operation.

In the majority of cases we style the GML data into an XML graphical format. The format doesn’t have to be XML-based, but given the increasingly wide availability of XSLT and XML-based graphics languages is at very least highly desirable. All of the examples in this article will assume that the target of GML Map Styling is an XML graphical format such as SVG, VML or X3D.

The process of transforming the graphical presentation (i.e. the output of the map styler in SVG, VML or X3D) into a viewable image we shall refer to as graphical rendering. At the present time there are a variety of graphical renders available for each of the different XML graphical formats, some native to the browser (e.g. Internet Explorer 5.0’s built in VML processor), some distributed as plug-ins for many browsers (e.g. Adobe SVG Viewer) and some available as stand alone viewers or code libraries (SVG and X3D). A Java applet SVG viewer is also available from Ionic Software.

Note that the steps in GML map making can readily be performed on either the client or the server. In fact, we will see that with only a very little work, one can make Map Style Sheets that are completely portable from client to client or from client to server.
The overall process of GML map making is illustrated in Figure 1.

The Map Styling function shown in Figure 1 might be realized in a variety of ways using many different languages and technologies. For our purposes, however, we are going to focus on Map Stylers that are based on XSLT Processors such as Xalan or Saxon, or the Java classes produced by an XSLT compiler. This enables us to use Map Style Sheets based on XSLT, which, as we shall see, allows us to create standard and portable Map Styles. With this assumption, a Map Styler is simply an XSLT Engine enhanced through the addition of a few standard extension functions.

4 Towards Standard Map Styles

When developing a standard Map Style or Map Style Sheet we use XSLT as our styling language. This is because:

- XSLT is a declarative language. You can read the style sheet and see what it is supposed to do without being concerned as to how the deed is accomplished.
- The output of the styling process is an XML graphical format. XSLT is well suited to XML-to-XML transformations.
- XSLT is extensible. XSLT supports the concept of an extension function. An extension function allows an XSLT engine to perform operations that are beyond the capabilities of the XSLT language such as geometric or coordinate
transformations. Furthermore, the XSLT extension function mechanism requires that the XSLT engine know only the signatures of the associated extension functions. This means that they can be standardized across all Map Stylers, providing truly portable map styling. Map Stylers can implement these extension functions as they choose, provided they agree to the Map Style interface definitions. We will have more to say on this in a moment.

What does an XSLT Map Style Sheet look like? Let’s look at a simple example. Elements in dark blue are GML or from the GML application schema.

Elements in red are from SVG.

```xml
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:ext="http://www.example.com/extension"
  version="1.0">
  <xsl:output method="xml" indent="yes" encoding="UTF-8"/>

  <xsl:variable name="extentOf" select="gml:extents/gml:extentsOf/gml:extent"/>
  <xsl:variable name="extentOfName" select="ext:extentOfName($extentOf)"/>

  <xsl:template match="gml:FLOODPLN_POLY/gml:featureMember">
    <xsl:apply-templates select="gml:FLOODPLN_POLY"/>
  </xsl:template>

  <xsl:template match="gml:FLOODPLN_POLY">
    <path fill="#cc0000" stroke="#cc0000" stroke-width="1"/>
    <path fill="#00cc00" stroke="#00cc00" stroke-width="1"/>
  </xsl:template>
</xsl:stylesheet>
```

As can be seen from this simple style sheet fragment, XSLT is a “search and replace” text manipulation language with special keywords for constructing XML output. The above style sheet fragment locates the `extentOf` (a GML geometry property) tag for each `FLOODPLN_POLY` feature (featureMember) in the GML dataset, and creates SVG graphical `path` elements for the inner and outer boundary elements of each located feature. The fill colour, stroke colour, and stroke width of the polygon boundaries are established in the style sheet fragment.

Note that this particular style sheet fragment also contains a test on a style sheet parameter to determine whether the style sheet output should be in world user coordinates or in pixel coordinates.

Now let’s look at another example. This one illustrates the use of extension functions:
This style sheet fragment locates the “coordinates” element in the GML dataset and uses the values in the coordinates element to construct an SVG path coordinate string using the extension function ‘convertCoordinatesToPath’. Note that the function is declared to the XSLT processor by the presence of the Extfun: prefix. The function converts the text contents of the coordinates element (given by the XSLT text() function) into the SVG path coordinate string with its embedded M, L plotter commands.

With a small number of such extension functions (which perform arithmetic and string operations that are difficult in XSLT), XSLT becomes a very flexible and powerful map styling language. Galdos Systems Inc., together with Ionic Software SA, has proposed a standard set of XSLT Extension Functions for Map Styling using SVG as the target graphical presentation language. The proposed style sheet functions were used in a recent OGC Pilot project. The draft specification for these GML Style Sheet Functions for SVG is available from Galdos Systems along with a Java jar file containing the extension functions themselves for the Saxon XSLT engine. The extension functions can be implemented in any language that can be accessed by the selected XSLT Style Engine, such as Java, Visual Basic and VBScript. Note that these extension functions, like the style sheets themselves, are specific to the target XML graphical format but that they are completely independent of the implementation language for the extension functions. Thus a single map styler can be readily constructed that can style into SVG, X3D and VML.

The style sheets examples above assume that the output of the styler is SVG. For more information on SVG see Adobe’s web site. There, you can download a free SVG plug-in for your Netscape or Microsoft browser as well as read an excellent tutorial and many interesting examples illustrating the power of SVG graphics.
5 Creating a Map Style Sheet

5.1 Getting Ready

Galdos Systems provides an easy-to-use Map Style Editor. Shown in Figure 2 below, this is a simple, graphical, internet-accessible application that you can use to create a GML XSLT Map Style Sheet for SVG.

Figure 2. The Galdos Map Style Editor

These are the requirements for using the Style Editor:

- You must have Microsoft’s Internet Explorer V5.x or Netscape 4.5 (or higher) installed on your machine.
- You must have the Sun Microsystems Java Virtual Machine (JVM) V1.2.2 installed in your browser. You can obtain a copy of the 1.2.2 JVM plug-in by going to the Sun Micro Systems web site (http://java.sun.com/products/plugin/) and downloading it. This is supplied as a self-installing executable. Simply run the downloaded file and follow the instructions.
The program will dynamically download the Ionic Software SVG rendering applet.

Now you need to set your Java security policy. This is defined in the java policy file located typically in Program Files\Javasoft\Jre\1.2\lib\security folder.

This is required in order to access Web Feature Servers that are on a machine other than that from which the Map Style Editor was retrieved. If in addition you wish to save Map Style Sheets to your local drive you must add the following additional lines to the java policy file:

```
grant codeBase "http://mapstyles.galdosinc.com/" {
    permission java.net.SocketPermission "*","connect";
};
```

To test your browser configuration, go to the Galdos Systems demonstration site at http://www.focalpoint.org/Xbed/VanMap.HTML and select a few feature layers (e.g. buildings, streets) and click the update button. A map of part of downtown Vancouver should appear in your browser window.

With that you are ready to roll!

5.2 Using the Map Style Editor

The process of creating a style sheet is shown in Figure 3.
The current version of the StyleEditor can be found at http://mapstyles.galdosinc.com/MapStyle.html.

Follow these steps to create a map style:
1. Locate a GML data source
2. Fetch the GML data source schema
3. Select a symbol library
4. Match symbols to feature types (e.g. “I want roads to be red lines”)
5. Click Create Style Sheet

More detailed instructions can be found at http://mapstyles.galdosinc.com/userdocs.html.

Now load the style sheet into your map styler and pass it the GML data you want portrayed as a map.

5.3 Getting GML Data
Some sources for GML data include:
GML data servers are being developed by a large number of geo-spatial vendors including CubeWerx Ltd., Ionic Software, ESRI, Oracle Corporation and Laser Scan. These provide for on-the-fly generation of GML data.

www.focalpoint.org/Xbed/VanMap.html

GML translation programs are in the works for the U.S. Census Tiger/Line format, as well as for most other popular GIS formats. These tools can be expected to be available in Q1/Q2 2001.

6 Putting it all together

The simplest way to try the Map Style Sheet Editor is to visit the GML central site at www.gmlcentral.com. There you will be able to:

1. Load style sheets from the style librarian
2. Apply the style sheet to a fixed data set and see the results of using different styles.
3. Create your own custom style sheets from scratch or modify one already in the Library.
4. Using the Map Style Editor you can create a wide variety of map styles. Each map style can be used to portray the same data in a wide variety of ways as shown in Figure 4. These simple maps were all generated from the same data source simply by changing the map style sheet.

Figure 4. Effect of Map Style Sheet on Map Appearance
GML is new and exciting technology that will drive the future of spatial information on the Internet. Take a test drive of the style editor and get a preview of where the web and geography are headed.