ELSEVIER

Computers & Geosciences 32 (2006) 184–194

COMPUTERS & GEOSCIENCES

www.elsevier.com/locate/cageo

The potential of XML encoding in geomatics converting raster images to XML and SVG

Byron Antoniou, Lysandros Tsoulos*

Cartography Laboratory, School of Rural and Surveying Engineering, National Technical University of Athens, Zografos Campus, Athens 15780, Greece

Received 1 February 2005; received in revised form 3 June 2005; accepted 3 June 2005

Abstract

The evolution of open standards and especially those pertaining to the family of XML technologies, have a considerable impact on the way the Geomatics community addresses the acquisition, storage, analysis and display of spatial data. The most recent version of the GML specification enables the merging of vector and raster data into a single "open" format. The notion of "coverage" as described in GML 3.0 can be the equivalent of a raster multi-band dataset. In addition, vector data storage is also described in detail through the GML Schemas and XML itself can store the values of a raster dataset, as values of a multi-table dataset. Under these circumstances an issue that must be addressed is the transformation of raster data into XML format and their subsequent visualization through SVG. The objective of this paper is to give an overview of the steps that can be followed in order to embody open standards and XML technologies in the raster domain. The last part of the work refers to a case study that suggests a step by step methodology to accomplish classification, an important function in Cartography and Remote Sensing, using the XML-encoded images.
© 2005 Elsevier Ltd. All rights reserved.

Keywords: Open standards; Raster image encoding; Classification

1. Introduction

The advent of XML-based technologies has gone beyond the expectations of the most optimistic users. Revolutionary ideas are emerging for the storage, exchange and display of data and new formats are created for almost all kinds of data, applications and knowledge domains. A considerable number of specifications have been issued by international organizations aiming at the provision of an efficient "open" environment to the user community.

Watching this frenzy trend of transforming everything into XML-based structures, one thing seems really out of the way: raster images. Languages like HTML and SVG do not have such a structural feature in their environment. Instead, both of them provide means (i.e., ⟨img⟩ or ⟨image⟩ elements) for the incorporation of raster images in text-based files, usually with an inline reference. The existing specifications instead of dealing with the problem bypass it and users treat raster formats more or less like a requisite tool for their work. The truth is that there is nothing common between XML-based structures and raster images. Raster image encoding is neither text based nor human readable and it cannot be parsed, checked for validity or well formedness. Moreover, the raster image content has

*Corresponding author. Tel.: +30 2107722730; fax: +30 210 7722734.

E-mail address: lysandro@central.ntua.gr (L. Tsoulos).

0098-3004/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cageo.2005.06.004

almost no flexibility (apart from resizing) in an XML-based environment, since pixel values are well locked inside the raster formats (Antoniou and Tsoulos, 2004).

Converting a raster formatted image into XML enables the user to utilize the information residing in the image and select, read and manipulate the parts of the XML file or the file as a whole in a number of ways in accordance with the application at hand.

This is the starting point for classification, statistical processing, filtering or the development of other applications with the use of XML technology. Operations like the storage and exchange of images acquire a new meaning in the framework of an interoperable environment like WebGIS. Converting an image from raster to SVG is even more exciting. An SVG-encoded image file enjoys all the above-mentioned advantages; in addition the user can visualize the effect of every change imposed on it.

Geomatics is a sector that depends on images to such an extent that one could tell that images are the most valuable geo-data sources. Feature extraction from a raster image is a very common task for geographical organizations around the world. Through the combination of image information and SVG code instead of vector data, digitization can produce scalable vector data (i.e., ⟨point⟩, ⟨line⟩, ⟨polyline⟩, ⟨polygon⟩ etc. elements) or GML (Geography Markup Language)-encoded data. Moreover, the need for visualization of the new components introduced by GML 3.0 specification, such as Grid functions, requires the rendering of continuous data in XML (Antoniou and Tsoulos, 2004).

Another area that will be influenced due to the evolution of open standards is interoperability. Provided that each satellite sensor stores data in its own format, the same applies to proprietary software, which also uses "closed" formats. The efficiency of remote sensing applications and interoperability can be enhanced with the use of open standards. By storing a raster dataset in an XML-encoded format, the information included in the dataset is conveyed intact along with the advantages that open standards bear. This paper elaborates on the technological environment, which can be utilized for the XML encoding of raster images and the extraction of information from the resulting datasets using open source methods.

2. Technological background

2.1. Extensible markup language—XML

XML stands for Extensible Markup Language and it is a W3C-endorsed standard for document markup. XML describes a class of data objects called XML documents and partially describes the behavior of computer programs that process them. XML documents are made up of storage units called entities, which contain either parsed or unparsed data. Parsed data are made up of characters forming character data or markup. Markup encodes the description of the document's storage layout and logical structure. Furthermore XML provides a mechanism to impose constraints on the storage layout and logical structure (Bray et al., 2000).

XML is a meta-markup language implying that it enables the user to create his/her own tags according to the application's needs. That means that XML does not have a fixed set of tags and elements that cover the needs of every application or user. Markup in an XML document describes the structure of the document along with the document's semantics. XML allows the developers to define properly the elements required and to encode their associations. XML defines the syntax that markup languages of each knowledge domain, such as MusicML, MathML, GML and SVG must follow. Although it is quite flexible in the elements' definition, it is rather strict in many other aspects. It provides grammar rules for the XML documents describing their proper structure, which allows for the development of XML parsers that can read any XML document. Documents that satisfy this grammar are considered as well formed (Elliotte and Means, 2002).

2.2. Geography markup language—GML

GML is a markup language used to encode and integrate spatial information, spatial relationships and non-spatial information, especially when non-spatial data are XML-encoded. GML also aims to serve both data transport and data storage, in a wide-area Internet context. GML exploits W3C standards to encode geographic information that can be readily shared in the Internet. In addition, GML provides a set of common geographic modeling objects to enable interoperability of independently developed applications. It is designed to support interoperability and does so by providing basic geometry tags, a common data model and a mechanism for creating and sharing application schemas. Although it is not the first meta-language introduced to describe geographic information, it is the first that has been widely accepted by the GIS community. An important characteristic of GML inherited from the XML specification is that it separates spatial and non-spatial content from presentation (Galdos, 2003).

2.3. Scalable vector graphics—SVG

SVG stands for Scalable Vector Graphics, an XML grammar for stylable graphics used as an XML namespace. SVG is a language for the description of two-dimensional graphics in XML and allows for the

encoding of three types of objects: vector graphics, images and text. Graphic objects can be grouped, styled, transformed and composed into previously rendered objects. The feature set includes nested transformations, clipping paths, alpha masks, filter effects and template objects, which are applied during rendering.

In general, SVG drawings can be interactive and dynamic. Animations can be defined and triggered either declaratively or via scripting (Ferraiolo, 2001). SVG graphics are scalable to different display resolutions. The same SVG graphic can be displayed at different sizes on the same Web page and re-used at different sizes on different pages. SVG graphics are scalable because the same SVG content can be a stand-alone graphic or can be referenced or included in other SVG graphics, thereby allowing a complex illustration to be built up in parts, perhaps by several people. During rendering SVG also provides client-side raster filter effects so that moving to a vector format does not result to the loss of popular effects such as soft drop shadows (Ferraiolo, 2001).

Other characteristics of SVG include a smaller file size and searchable text information. An SVG file—utilizing the elements provided by the specification—is usually smaller than a raster file for the same map resolution and thus can be transferred across the Internet more quickly. Text information inside SVG is still text and can be searchable, while text information inside the raster file becomes integrated into the image and is no longer recognized as text. SVG is also particularly suitable for displaying intelligent maps, because geometric objects such as points, lines, and polygons are recognized as such and are identifiable.

Raster images on the other hand contain information about every pixel, and points, lines and polygons that are no longer recognizable. Therefore, the user can directly work with spatial features on an SVG but not on a raster graphic image (Peng and Zhang, 2004).

SVG is also based on XML and therefore conforms to other XML-based standards and technologies, such as XML Namespace, XLink, and XPointer. XLink and XPointer allow for linking from within SVG files to other files on the Web, like a GML data element, HTML pages or other SVG files (Boye, 1999).

2.4. Extensible stylesheet language transformation—XSLT

XSLT is a language that enables the user to convert XML documents into other XML documents or into almost any form an application or a user needs. XSLT provides an easy, W3C sanctioned, way to convert XML documents that conform to a schema into documents that conform to another, enabling the sharing of information between different systems. From another perspective XSLT is a programming language that describes the way and the methods to be followed for the transformation of a well-formed tree structure of an XML document to another. XSLT is not the only way to achieve these goals; there are alternative ways to transform XML documents but XSLT has prevailed. The fact that XSLT is a W3C standard implies that XSLT complies with the specifications published by the W3C or those that will be announced in the future (Clark, 1999). Furthermore, the non-proprietary status of the specification and the platform independency, guarantee the prospects and the integrity of the specification. In addition, the fact that the XSLT document transformation instructions are stored as an XML document is an advantage since there is no need for using another syntax. The above-mentioned characteristics justify the popularity of XSLT when it comes to XML transformation (DuCharme, 2001).

2.5. Data model encoding

The accurate representation of the complexity of the real world has been one of the fundamental problems in Geomatics. In order to address this issue the GIS community has introduced a number of ways for modeling real world data.

In a vector model, reality is perceived as an aggregation of discrete entities defined by their geometry, topology and thematic attributes. Another popular way to describe real world phenomena is by using raster structures that can sufficiently describe continuous/field data. The raster model in its basic form is considered as a rectangular array of equally spaced pixels. Each pixel can be defined uniquely inside the array through i,j coordinates. The values stored in the pixels of the array depend on the phenomenon and can be from elevation or temperature to the reflectance of light for part of the spectrum.

2.5.1. The XML approach

XML enables domain experts to create properly structured formats, which can serve the storage and exchange of a wide variety of data types. Apart from that XML itself can store efficiently various types of data and it can easily describe continuous data using a table-based mapping. The table-based mapping is used to model XML documents as a single table or set of tables (Fig. 1).

This general form can be modified accordingly in order to accommodate the description of different phenomena. For instance the description of a three-band raster image could be achieved in the way shown in Fig. 2. A problem easily solved through XML is whether the auxiliary data should be stored as child elements or as attributes, as well as the names to be used for each element or attribute. In addition, developers who use table-based mappings often include table and column

```
<tableholder>
  <table1>
    <row>
      <column1>...</column1>
      <column2>...</column2>
      ...
    </row>
    <row>
      ...
    </row>
    ...
  </table1>
  <table2>
    ...
  </table2>
  ...
</tableholder>
```

Fig. 1. Table-based mapping of field data.

```
<image>
 <band1>
      <pixel id="0" row="0" column="0">
        <r>201</r>
        <g>171</g>
        <b>81</b>
    </pixel>
      <pixel id="1" row="0" column="1">
        <r>203</r>
        <g>175</g>
        <b>78</b>
    </pixel>
    <pixel>
    .....
    </pixel>
    .....
 </band1>
 <band2>
    .....
 </band2>
 <band3>
    .....
 </band3>
</image>
```

Fig. 2. Description of a three-band raster image.

metadata either at the beginning of the document or as attributes of each table or column element. For image metadata one could store pixel size, date and conditions of capture, details of the camera used, etc. Using this method efficient description, storage and exchange of field-type geographic data can be achieved utilizing open standards and promoting interoperability (Bourret, 2004).

Apart from continuous data, XML can easily encode data modeled as vectors. An XML schema is developed describing the definitions of geometric primitives. In fact that is what OGC provides through GML, which is an XML-based specification designed to describe vector data.

2.5.2. The GML approach

GML has been a turning point in Geomatics to the extent that many national and private organizations have already adopted this format. GML 3.0 specification introduced a number of new components improving vector data encoding (Cox et al., 2003). GML 2.1 provides only three core schemas (features.xsd, geometry.xsd and xlinks.xsd) whereas in GML 3.0 there are 28 core schemas.

This version supports new geometry types including *Arc*, *Circle*, *CubicSpline*, *Ring*, *OrientableCurve*, *OrientableSuface*, *Solid,* and the aggregates *CompositeCurve*, *CompositeSurface* and *CompositeSolid*. GML 3.0 addresses many limitations of the previous version like: topology, temporal components, dynamic features, coverages and coordinate reference systems.

These components enable the encoding of sophisticated vector models, which are more accurate representations of the real world. Furthermore, GML provides XLink and XPointer mechanisms to make geospatial data interoperable. Through XLink and Xpointer, different features and feature collections, which may be located remotely, can be associated at the feature level (Peng and Zhang, 2004).

Apart from the improvements in vector encoding, GML 3.0 introduced a new way for the description of continuous data with the use of "coverages" (Fig. 3). The specification defines the GML encoding for coverages and is compliant with the conceptual model of ISO 19123:

> Coverages support mapping from a spatiotemporal domain to attribute values where attribute types are common to all geographic positions within the spatiotemporal domain. A spatiotemporal domain consists of a collection of direct positions in a coordinate space. Examples of coverages include rasters, triangulated irregular networks, point coverages, and polygon coverages (Cox et al., 2003).

There are basically two methods to describe information in a coverage. The first is by creating a set of discrete location–value pairs. The domain set of the coverage is formed from discrete and homogeneous

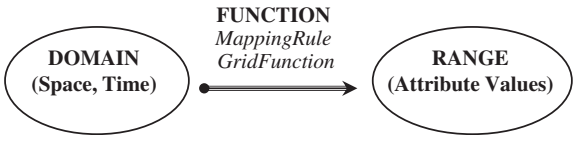


Fig. 3. Conceptual model of a Coverage.

feature collections and the range set is formed by a set of values. The properties of each feature are assigned to the location of the feature. The second method describes the spatio-temporal domain set and the set of values from the range set. It also describes the rule that assigns a value from the range set to each position within the domain (Cox et al., 2003). Two schemas, coverage.xsd and grids.xsd support coverages in GML 3.0 and the values that can be assigned in the domain set are *Grid* and *RectifiedGrid* as shown in Figs. 4 and 5.

The ⟨*gml:Grid*⟩ element defines an non-rectified grid, which is a network composed of two or more sets of equally spaced parallel lines intersecting each other.

The ⟨gml:RectifiedGrid⟩ defines another kind of grid. The points of a rectified grid refer to geographic locations. The element describes the position of the origin of the grid along with the position of the offset vectors that determine the spacing of the points of the grid. In the above example the RectifiedGrid starts at the origin (580000, 4500000) and the spacing is 10 units.

```
<gml:Grid dimension="2">
        <gml:limits>
                <gml:GridEnvelope>
                        <gml:low>0 0</gml:low>
                        <gml:high>5 5</gml:high>
                </gml:GridEnvelope>
        </gml:limits>
        <gml:axisName>u</gml:axisName>
        <gml:axisName>v</gml:axisName>
</gml:Grid>
```

Fig. 4. An instance of a simple Grid.

2.5.3. The SVG approach

As already mentioned, both XML and GML can encode the most common methods to represent real world, utilizing vector and field models. In Geomatics, visualization is as important as encoding. It does not make sense to encode geometric features without displaying them on a map. This is undertaken by SVG, which is an XML-based visualization tool. As implied by its name, SVG was designed for the visualization of vector data. The SVG 1.2 working draft (Jackson, 2004) enriches the content of the previous specifications allowing for the visualization of vector data in various ways. Elements like *circle*, *rect*, *ellipse*, *line*, *polyline*, *path*, *textpath* and *text,* which describe geometry can be combined with elements that define gradients, filters, animation, linking, scripting, flowing text, streaming, progressive rendering, vector effects, audio, video, etc.

Although SVG specification defines an *image* element, which can be used to embody a raster image into an SVG file, it is evident that there is nothing common between XML-based structures resulting from the SVG specification and raster images. Raster images are neither text based nor human readable. They cannot be parsed, checked for validity or well formedness. Moreover, the raster image content has almost no flexibility (apart from resizing) in an XML-based environment, since pixel values are well locked inside the raster formats. In order to utilize the information that a raster image holds, a reconstruction is needed as shown in Fig. 2 (Antoniou and Tsoulos, 2004).

```
<gml:Rectified Grid dimension="2">
        <gml:limits>
                <gml:GridEnvelope>
                        <gml:low>580000 4500000</gml:low>
                        <gml:high>600000 45          20000</gml:high>
                </gml:GridEnvelope>
        </gml:limits>
        <gml:axisName>x</gml:axisName>
        <gml:axisName>y</gml:axisName>
        <gml:origin>
                <gml:Point gml:id="orig1">
                        <gml:coordinates>580000,          4500000</gml:coordinates>
                </gml:Point>
        </gml:origin>
        <gml:offset      Vector>10, 0</gml:offsetVector>
        <gml:offsetVector >0, 10</gml:offsetVector>
</gml:RectifiedGrid>
```

Fig. 5. An instance of a simple RectifiedGrid.

3. Analysis

Images constitute the most valuable sources for geographic data acquisition and the extraction of information from raster datasets is carried out with the use of commercial software. Processes like image preprocessing, enhancement, analysis, classification, vectorization and visualization are performed in commercial environments since there are no corresponding freeware applications.

In order to work in an "open" environment for the creation of SVG or GML elements from raster datasets, we should follow the alternative paths/ways shown in Fig. 6. The first step is the transformation of the raster image to XML encoding. This can be achieved with the use of any programming language, which can easily decompose three-band raster datasets. However, when it comes to multi-band datasets, data from all bands must first be saved in an ASCII file and then in XML. Provided that XSLT is ideal in manipulating XML documents, XSLT-templates can be built for the transformation of XML into classified XML datasets based on pixel values and remote sensing elements. The resulting document can be used either for the generation of GML data stored with the Grid functions or for the creation of SVG image files rendered on a cell by cell basis.

The SVG image files can be further processed with XSLT exploiting the SVG specification components (i.e. transformations, matrices, etc.), for image processing or for the creation of more sophisticated data like geo-referenced or ortho-rectified SVG image files. The SVG image files can be transformed into GML data or SVG elements. This is implemented through a raster to vector transformation bearing in mind that we do not deal with raster images but with a text file that describes explicitly a raster dataset. In addition, at any point of the process ECMAScript can be used for SVG element digitization and subsequent transformation into GML entities using XSLT.

4. Application

A case study was carried out in order to verify the efficiency of the method and evaluate the results. The application elaborates on the ways and methods used to achieve a truthful and accurate outcome, which is subsequently compared with the results taken with the use of proprietary software.

The dataset used is a subset of a Landsat TM scene showing an island. The raster dataset has seven bands with 25 m ground resolution (164 × 190 pixels). Only six bands were used from the dataset since the sixth band is

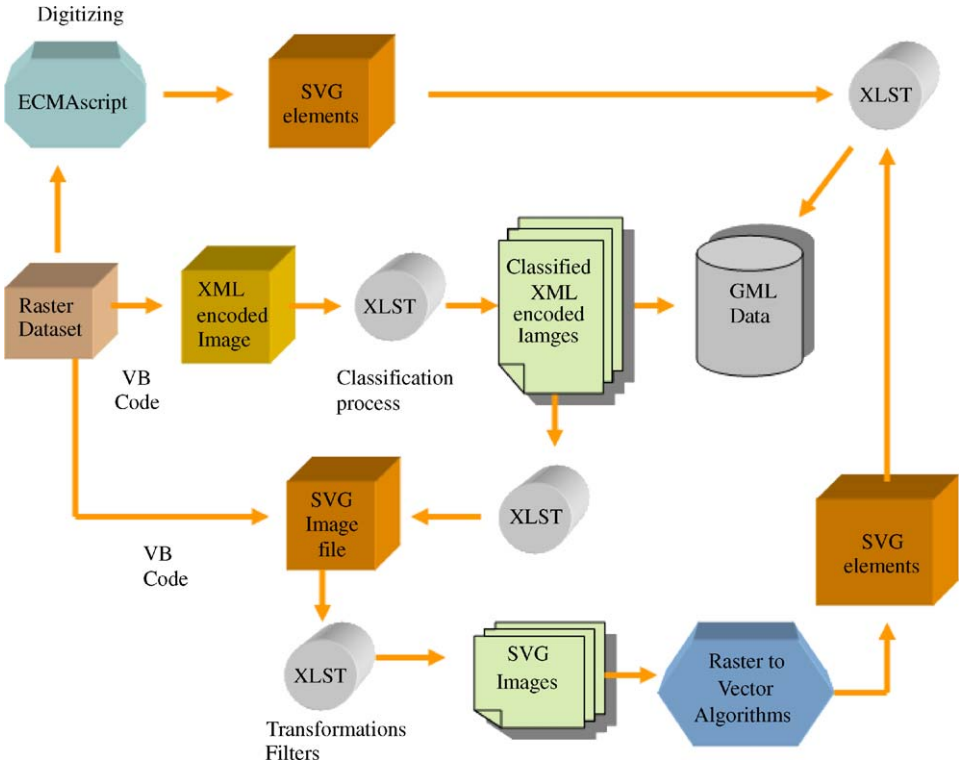


Fig. 6. Alternative ways for creation of SVG or GML elements from a raster dataset.

the thermal band and it is not normally used in a classification process. Each band is stored as a different file in tiff format. Pixel values in these files range from 0 to 255 in grayscale. Fig. 7 shows a colored composite of the original scene.

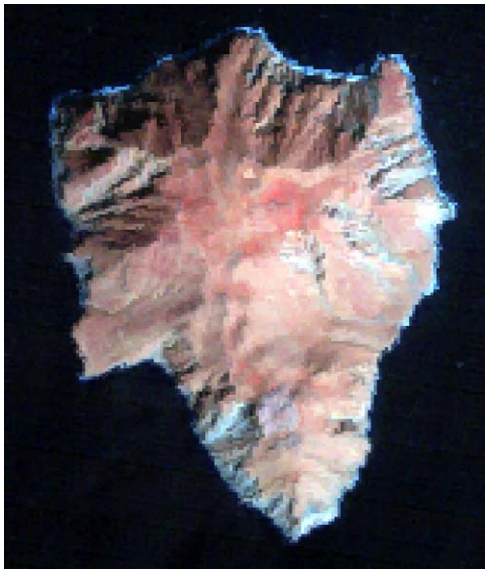


Fig. 7. Original scene.

The original data were transformed into the form shown in Fig. 8A and stored as an ASCII file. The file records the x, y position of each pixel in the grid and the corresponding values of the bands of the dataset (except the thermal one). Alternately, in the structure shown in Fig. 8B the positions of each pixel in the grid is replaced by the true position of the ground element.

4.1. XML–SVG transformation

A short program in Visual Basic transforms the ASCII file in a well-formed and valid XML format. The file is validated through a predefined XML schema, which can support the storage of a multi-band raster dataset. In the future this XML file and the corresponding schema could be supplied directly by the provider of the remotely sensed data. In other words, the initial data could be an XML file including the data and the metadata of the scene along with the XML schema, which will be the reference for each scene instance of each satellite sensor. A sample of the XML file is shown in Fig. 9.

Metadata storage is an issue that must be considered depending on the application. Metadata about the remotely sensed image may include a variety of information ranging from pixel size and the number of rows and columns, up to the condition of the sensor

X	Y	B1	B2	B3	B4	B5	B7
0	0	71	21	17	9	6	5
1	0	70	21	17	9	7	6
2	0	71	21	16	9	6	5
3	0	71	21	16	9	6	5
4	0	70	21	16	9	7	5
5	0	72	22	17	10	8	5
6	0	70	22	17	9	7	4
7	0	70	21	17	9	10	5
8	0	74	22	17	10	6	4
9	0	73	22	17	10	9	8
10	0	72	22	18	11	9	3
11	0	70	21	17	10	7	6

(A)

X	Y	B1	B2	B3	B4	B5	B7
518942.000	4073488.000	71	21	17	9	6	5
518967.000	4073488.000	70	21	17	9	7	6
518992.000	4073488.000	71	21	16	9	6	5
519017.000	4073488.000	71	21	16	9	6	5
519042.000	4073488.000	70	21	16	9	7	5
519067.000	4073488.000	72	22	17	10	8	5
519092.000	4073488.000	70	22	17	9	7	4
519117.000	4073488.000	70	21	17	9	10	5
519142.000	4073488.000	74	22	17	10	6	4
519167.000	4073488.000	73	22	17	10	9	8
519192.000	4073488.000	72	22	18	11	9	3
519217.000	4073488.000	70	21	17	10	7	6

(B)

Fig. 8. (A) Sample ASCII file of original data. (B) Sample ASCII file of original data with true positions.

```
encoding='UTF-8'?>
<image>
<pixel id='0' row='0' column='0'>
<b1>71</b1>
<b2>21</b2>
<b3>17</b3>
<b4>9</b4>
<b5>6</b5>
<b7>5</b7>
</pixel>
<pixel id='1' row='1' column='0'>
<b1>70</b1>
<b2>21</b2>
<b3>17</b3>
<b4>9</b4>
<b5>7</b5>
<b7>6</b7>
</pixel>
<pixel id='2' row='2' column='0'>
<b1>71</b1>
<b2>21</b2>
<b3>16</b3>
<b4>9</b4>
<b5>6</b5>
<b7>5</b7>
</pixel>
...
...
</image>
```

Fig. 9. Sample of XML file.

during the capture of the scene and orbital data. It can be stored in the same file with the pixel values or in a separate file using the XML format. Both ways are acceptable since XSLT is able to manipulate multiple documents (i.e. data and metadata file) as input.

Following the same procedure, the ASCII file can be transformed directly to SVG image file. This structure of the SVG image file can take advantage of the nested structure supported by the SVG specification and nest all bands in a single SVG file. Since it is possible to show only three bands at a time, the choice of the bands to be shown along with their order (color/band correspondence) could be decided by the user with a set of predefined widgets written in ECMAscript. This is the equivalent of creating pseudo-colored images with proprietary remote sensing software.

4.2. Classification

The result of the process described in 4.1 can be transformed into a classified XML file again with the use of XSLT. A number of XSLT templates are applied depending on the entities to be classified. More specifically, the objective is to classify vegetation, waste land and water regions using two normalized band ratios. The first band ratio discriminates water from land and the second is a vegetation index. These ratios take into consideration the values of two bands (1 and 5 the first and 3 and 4 the second) of each pixel and assign to it the new value that is calculated through the following formulas:

$$\text{Sea\_indicator} = (b1 - b5)/(b1 + b5), \tag{1}$$

$$\text{Vegetation/Waste land\_indicator} = (b4 - b3)/(b4 + b3). \tag{2}$$

The second formula is widely known as Normalized Difference Vegetation Index—NDVI (Lillesand and Kiefer, 2000). The outcome of this procedure is an XML-encoded file, which describes an array of pixels whose values result from formulas (1) and (2). Both formulas are inserted as templates in a XSLT stylesheet (Fig. 10), which transforms the XML-encoded image file into a classified one (Fig. 11).

The most efficient way to visualize geographic information stored in XML is through SVG. The XSLT file includes a template that supports one of the above-mentioned comparisons. The comparison results in a true-or-false state and accordingly a black or white value is assigned to each pixel position resulting in a "binary" SVG file that has a ⟨rect⟩ element for each pixel filled with black or white color. The size of the rectangle can be set either to 1×1 pixels or to 25×25 m (the actual size of the corresponding ground element) depending on the requirements of the application. This process results in a classified SVG image file, which differentiates sea (black—background) from land (white—foreground), as shown in Fig. 12A. Fig. 12B shows the outcome of the classification resulting through ERMapper, which is identical with the SVG-encoded file. In order to achieve the classification of the sea/land the same steps are followed. It is pointed out that these methods do not constitute a sophisticated implementation of remote sensing techniques. They only prove the concept of applying basic remote sensing and image processing principles utilizing open standards and XML-based technologies.

A subset of the classified XML image file is shown in Fig. 11.

If sea_ind > 0.67 the pixel corresponds to the sea, otherwise it denotes land. In the same way if the vw_ind > 0.3 then the pixel belongs to vegetation.

4.3. Raster to vector conversion

The conversion of the SVG image file into vector may sound awkward since the elements contained in a SVG file are already vectors. Well, in this case a vector element (the ⟨rect⟩ element) is used to describe the data of a raster dataset. In other words the ⟨rect⟩ element

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" version="1.0" encoding="UTF-8"
indent="yes"/>
<xsl:template match="/">
<image>
  <xsl:for-each select="/image/pixel">
    <pixel>
        <xsl:attribute name="id">
            <xsl:value-of select="@id"/>
        </xsl:attribute>
        <xsl:attribute name="row">
            <xsl:value-of select="@row"/>
        </xsl:attribute>
        <xsl:attribute name="col">
            <xsl:value-of select="@column"/>
        </xsl:attribute>
        <sea_ind>
            <xsl:value-of select="substring((b1 - b5) div (b1 +
     b5),1,5)"/>
        </sea_ind>
        <vw_ind>
            <xsl:value-of select="substring((b4 - b3) div (b4 +
     b3),1,5)"/>
        </vw_ind>
    </pixel>
</xsl:for-each>
</image>
</xsl:template>
</xsl:stylesheet>
```

Fig. 10. Creation of band ratios with XSLT.

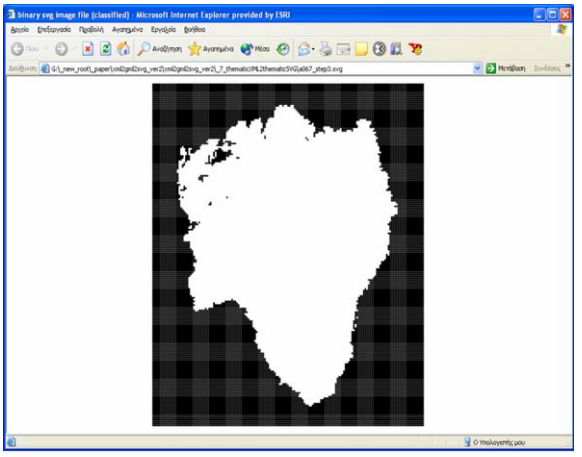
```
<?xml version="1.0" encoding="UTF-8"?>
<image>
<pixel id="0" row="0" col="0">
<sea_ind>0.844</sea_ind>
<vw_ind>-0.30</vw_ind>
</pixel>
<pixel id="1" row="1" col="0">
<sea_ind>0.818</sea_ind>
<vw_ind>-0.30</vw_ind>
</pixel>
<pixel id="2" row="2" col="0">
<sea_ind>0.844</sea_ind>
<vw_ind>-0.28</vw_ind>
</pixel>
...
...
</image>
```

Fig. 11. Sample of classified image file.

bridges the raster and the vector. Information encoded in the SVG file has to be decoded and described with the use of geometric features. This can be implemented either manually or semi-automatically.

In the manual way, the user will digitize over the SVG image the features that the classification revealed (i.e. coastline). In order to accomplish that, a full set of predefined widgets must be created with the use of ECMAscript. The set must include tools for the digitization of geometric features (point, line, polyline, etc.) as well as tools for their manipulation (delete, edit, save, etc). Ideally, the ECMAscript widget could look like a CAD toolbox (including the manipulation of layers). The result of digitization will be a set of SVG geometric elements, which can be used either directly for the composition of SVG maps or can be transformed with the use of XSLT into GML format and then stored in a database. In the semi-automatic conversion, the "binary" SVG file is transformed into geometric SVG or GML elements with the use of software modules like the ones mentioned in Battiato et al. (2005) that embody SVG technology and a programming language.

Another option is to take advantage of the new capabilities that GML 3.0 offers for the storage of continuous/field data, using GML "coverages". The conversion of the classified SVG image file into a GML coverage is carried out with the use of XSLT. From the column and row attributes of the ⟨pixel⟩ elements the description of the grid can be inferred and the domain set of the grid can be defined accurately. The values of the pixels can be stored in the range set of the GML



(A)

(B)

Fig. 12. (A) Classification through XML technology. (B) Classification through ERMapper.

coverage and subsequently assigned to the domain set through a simple mapping rule.

5. Performance

The standard-based GML–SVG approach has implications on performance. GML data are text based, therefore it is easier to be transported over the network than images. But when GML-coded geospatial data are transported, all the mark-up elements that describe spatial and non-spatial features, geometry and spatial reference systems of the data are also transported to the recipient. This is important for data interoperability, because the GML-coded data could be saved and used by any other client side applications that can read GML data. One shortcoming of using GML as a means of transporting data is its size. The size of the resulting GML and SVG data files compared with the GIS data formats, is large and it is related to map resolution (Peng and Zhang, 2004). By reducing the numerical precision from millimeter to centimeter resolution, a considerably smaller number of vertices are necessary to describe the vectors and the GML and SVG file size is reduced drastically.

As pointed out earlier, with GML and SVG data files the user can always get the full resolution map no matter how closely the user zooms in anywhere on the SVG map. On the contrary, with raster image files the image becomes "blurred" when the user zooms in to the details of the map image. The basic means to improve the performance of SVG and GML is to compress GML and SVG data files. They are text based and thus much easier to compress than other formats. Other compression approaches include line interlacing or reordering of pixels (columns/lines).

6. Conclusions—future work

This paper elaborates on the XML encoding of raster images and their utilization in an open environment. The work carried out so far shows that the combination of the three standard technologies (GML, SVG, and XSLT) has a great potential to address a number of issues. GML is an effective means for the encoding, storage and transport of geospatial data. SVG produces high-quality graphics, which is ideal for displaying spatial data and making intelligent maps.

The main advantages of GML, SVG and XSLT are that they are open standards and that they are based on XML. Therefore, they can work with other XML technologies and other standards like Simple Object Access Protocol (SOAP), something that makes GML and SVG efficient means for delivering, accessing and processing of geospatial data. However, there are still many issues to be resolved before this can be accomplished. More research and experiments are needed to optimize and test algorithms used for the XML raster encoding—classification and the compression of the resulting files.

References

Antoniou, B., Tsoulos, L., 2004. Converting raster images to XML and SVG—The Potential of XMLencoded Images and SVG Image Files in Geomatics. SVGOpen 2004, Tokyo, Japan. http://www.svgopen.org/2004/papers/Converting_raster_images_to_XML_and_SVG/>

Battiato, S., Blasi, G., Gallo, G., Nicorta, S., Messina, G., 2005. SVG Rendering of Digital Images: an Overview. In: WSCG' 2005. International Conference on Computer Graphics, Visualization and Computer Vision. Plzen, Czech Republic.

http://wscg.zcu.cz/wscg2005/Papers_2005/Poster/G19-full.pdf>

Bourret, R., 2004. XML and Databases. http://www.rpbourret.com/xml/XMLAndDatabases.htm>

Boye, J., 1999. SVG Brinks fast Vector Graphics to the Web. http://www.ddj.com/documents/s=3213/nam1012432601/index.html>

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., 2000. Extensible Mark-up Language (XML) 1.0, second ed. W3C Recommendation. http://www.w3.org/TR/2000/REC-xml-20001006>

Clark, J., 1999. Extensible Stylesheet Language Transformations (XSLT), W3C Recommendation. http://www.w3.org/TR/1999/REC-xslt-19991116>

Cox, S., Daisey, P., Lake, R., Portele, C., Whiteside, A., 2003. Geography Markup Language (GML) Implementation Specification. http://www.opengeospatial.org/docs/02-009.pdf>

DuCharme, B., 2001. XSLT Quickly. Manning Publication Co., Greenwich, CT 298pp.

Elliotte, R.H., Means, W.S., 2002. XML in a Nutshell, second ed. O'Reilly, Sebastopol, CA 613pp.

Ferraiolo, J., 2001. Scalable Vector Graphics (SVG) 1.0 Specification. W3C Recommendation. http://www.w3.org/TR/2001/REC-SVG-20010904/>

Galdos Systems Inc, 2003. Developing and Managing GML Application Schemas. http://www.geoconnections.org/developersCorner/devCorner_devNetwork/components/GML_bpv1.3_E.pdf>

Jackson, D., 2004. Scalable Vector Graphics (SVG). http://www.w3.org/TR/2004/WD-SVG12-20040510/>

Lillesand, T.M., Kiefer, R.W., 2000. Remote Sensing and Image Interpretation, fourth ed. Wiley, New York, NY 724pp.

Peng, Z.R., Zhang, C., 2004. The roles of GML, SVG and WFS specifications in the development of Internet GIS. Journal of Geographical Systems 6, 95–116.