Web Mapping and WebGIS: do we actually need to use SVG?

Vyron Antoniou

PhD Student
UCL DEPARTMENT OF CIVIL, ENVIRONMENTAL AND GEOMATIC ENGINEERING

London
Gower Street
WC1E 6BT
UK

Jeremy Morley

Lecturer in Geographic Information Systems
UCL DEPARTMENT OF CIVIL, ENVIRONMENTAL AND GEOMATIC ENGINEERING

London
Gower Street
WC1E 6BT
UK

Abstract

Since the World Wide Web became a medium to serve spatial information there have been different methods to deliver a map over the Web. These methods vary from a trivial use of HTML's element to highly complex and sophisticated ones like distributed GIS services. Three of these methods are going to be briefly discussed due to their

roles in the formation of Web mapping and Web GIS knowledge domain. These approaches are: (i) a methodology, followed by major commercial vendors, which rely on on-the-fly generation of raster maps on the server; (ii) an architecture proposed by OGC, a leading international organization, and introduces the concept of Web mapping services; and (iii) a solution based on popular APIs from companies like Google, Microsoft and Yahoo, that is considered as a start of a new era in Web mapping.

While these solutions have managed to deliver spatial content to the users, they are heavily based on the pre-rendering of spatial information in a raster format. Such a strategy, though, bears a number of limitations. Limitations such as content inflexibility, limited interactivity and animation, limited functionality in the applications, non-conformance to a bi-directional Web, make this strategy not fit-for-purpose when it comes to special applications and visualization of XML-based data. This will be examined in order to justify the need for vector data on the client.

While there is a growing need of vector encoding on the client, the fact remains that still there has not been a widespread acceptance and progress of mapping applications that use SVG though there are alternatives through hybrid mapping applications.

A number of limitations are associated with vectors that range from intrinsic characteristics of the encoding to corporate decisions that should provide a focus for scholars and developers. The size of the XML-based files, the difficulties in holistically determining and modelling the cartographic process of generalization and unresolved issues related to efficient progressive transmission of vector data are major drawbacks. Moreover, the open structure of XML encoding raises intellectual property protection and rights management issues when serving spatial information. Finally, the denial of native support from IE, the predominant web browser, and the discontinuity in development from Adobe of the most popular plug-in deters the adoption of SVG.

In the context of the aforementioned theoretical background a series of real world examples is examined. Vendor APIs of mapping applications are examined and the role of SVG in these is evaluated. Additionally, the effort to build task oriented applications like routing services (from vendors like Google, Yahoo and Microsoft) using SVG

is discussed. Also, the realisation of bi-directional Web applications and the rise of volunteered geographic information is examined using wikimapia.org as a paradigm. Finally, a case study of delivering legislation, planning and building regulation information with the help of a WebGIS will be presented. This application was built using PostgreSQL (with PostGIS) as the spatial database and SVG to deliver maps.

After the first wave of enthusiasm for the new mapping era based on popular APIs that led to the boom of mash-ups, the need for the next step in the evolution of Web mapping applications has emerged. It is clear that a raster only solution, though capable of providing an overwhelming amount of information, has a number of limitations due to the nature of raster files and thus vector overlays are needed to cover these limitations. In an effort to determine the most promising way for further research in vector transmission, a set of preliminary tests were undertaken in order to evaluate the efficiency of a hybrid mapping application using SVG, AJAX and the Google Maps API. These experiments show the inadequacy of Google Maps API to provide ways for efficient transmission of vector data. On the contrary, the use of Google Maps as a backdrop overlaid with spatial data encoded in SVG proves to be considerably more dynamic and efficient solution. Moreover, in these examples a new approach for the use of AJAX techniques in the transmission process of vector data is introduced that takes advantage of the particularities of SVG grammar. The characteristic of this method is the minimization of network latency and thus smooth rendering of vector data on the client without delays or "loading…" messages.

Table of Contents

Introduction

Since the World Wide Web (Web) became a medium to serve spatial information there have been different methods to deliver maps over the Web. These methods vary from the delivery of maps as static files to highly complex and sophisticated distributed GIS services (Peng and Tsou 2003). Delivery of maps is heavily based on the pre-rendering of spatial information in a raster format. This leads to limitations such as content inflexibility, low interactivity and animation, non-conformance to a bi-directional Web, and others. These limitations make this strategy not fit-for-purpose when it comes to advanced applications and the visualisation of XML-based data. On the other hand, although there is a growing need of vector encoding on the client, there has not been a widespread acceptance of hybrid mapping applications and the use of vector formats such as Flash, VML or SVG is far from extensive.

Today, the advent of new services such as Google Maps, Yahoo Maps or Microsoft Virtual Earth and the release of their APIs, sparked the map mash-ups phenomenon presenting the field of Web mapping and Web GIS with new challenges. After the first wave of enthusiasm for the new mapping era based on popular APIs, limitations still remain and the need for the next step in the evolution of Web mapping applications has emerged. It is clear that a raster only solution, though capable of providing an overwhelming amount of information, has a number of limitations due to the nature of raster files and thus the need for vector overlays to cover these limitations is more demanding.

The paper starts with a discussion about the current situation in Web mapping and Web GIS, and presents the predominant forces of the field which are heavily based in raster format. In Section 3, the discussion continues by presenting the limitations that raster-only solutions impose. Several real-world *cases are presented as examples in order to highlight these limitations. In parallel, the focus is turned on SVG since most of these limitations can be by-passed with the help of SVG. Next, the basic limitations from the use of vector data are discussed. Section 5 addresses the issue of data transmission and a new approach of delivering spatial content over the Web by using intrinsic characteristics of SVG grammar is presented. Conclusions are drawn in Section 6.*

Current Situation in Web Mapping

Throughout the evolution of Web mapping, delivery of maps was heavily based on the pre-rendering of spatial information in a raster format. By far the most important factors for the penetration of the raster-only methods in Web mapping is the nature of core Web technology. Raster data is one of the main components of Hyper Text Markup Language (HTML), the specification used to publish content on the Web and these methods exploit the native capability of Web browsers to render common raster formats (GIF or JPEG). This means that accessing a raster-based map is a straightforward process (ESRI 2006), since there is no need to install external plug-ins, as is common with vector formats, and consequently there are no obstacles to access the map, such as need for advanced functions or security concerns associated with downloading and installing software.

The discussion continues with the most important representatives of raster-based Web mapping and Web GIS efforts.

Professional Web-GIS software

In the early days, the expansion of cartography in the Web leaded to the convergence of cartography and GIS in order to generate on-the-fly maps according to user's criteria. These efforts were originally supported by companies traditionally active in the domain of desktop GIS by building dedicated Internet Map Servers (IMSs). The main characteristic among such commercial software is the generation of raster image maps upon request at the server by IMS. The mapping application transformed user's requests into criteria for extracting spatial content from the server. The extracted data is styled properly and rasterised into a single image file which is then transmitted to the client and rendered on the user's browser (Plewe 1997). This strategy is followed by most specialized commercial GIS and mapping applications up to *now. The major advantage of this method is the efficiency and scalability of the IMS. An IMS can potentially have the same functionality that a desktop GIS application has and thus can offer a variety of GIS tools and functions. Additionally, custom functions can be easily built at any time. Moreover, GIS companies built IMSs as continuations to their desktop GIS software and thus work harmonically with the rest of the of-line cartographic procedure.*

Open Geospatial Consortium (OGC)

The second delivery method evolved from the effort to tackle the problem of interoperability and to provide the means for easy access to spatial information scattered over the Web. The OGC is leading an effort to build a broad framework for Web mapping services and supports this through the development of open specifications and standards. In this method, the user's request is transmitted to an OGC compliant server which either is the sole source of information or capable of propagating the request to other OGC compliant servers. The answer of each responding server is generated at the time of the request and sent to the user. The advantage of this method, which is based in open standards and interoperability, is the ability to acquire spatial content from different sources and present the augmented outcome to the user.

Tile Servers, AJAX and Application Programming Interfaces (APIs)

Finally, the third method is the most efficient practice of Web-mapping and it is based on the use of pre-generated fairly small raster tiles that compose the final image which is presented to the end-user. In this method, the map is pre-rendered at all possible zoom levels that will be provided to the user. This results in very large raster image maps which are then cut into small tiles and stored on the server. Although this can lead to a huge number of tiles for a detailed or large dataset, storing, indexing and handling of raster files is straightforward. The tiles are loaded into the Web browser window as a matrix, and from the user's perspective it seems as a continuous image that comply with the limited set of parameters passed by the user. For any consequent request such as pan, zoom or managing layers a new request is made by the application that runs on the client's Web browser usually using AJAX (Asynchronous Javascript and XML) techniques and hence the server transmits to the client only the tiles that are needed in addition to the ones currently in the client's cache. This method makes the application considerably faster and responsive in comparison to the previous methods *since eliminates the need to extract data and rasterize the map since the tiles are already raster files. Major commercial providers of public mapping like Google, Microsoft and Yahoo adopted this method in their Web mapping applications while at the same time offered powerful and easy to use APIs. Today, the advent of new services such as Google Maps, Yahoo Maps or Microsoft Virtual Earth and the release of their APIs, sparked the map mash-ups phenomenon presenting the field of Web mapping and Web GIS with new challenges. After the first wave of enthusiasm for the new mapping era based on popular APIs, limitations still remain and the need for the next step in the evolution of Web mapping applications has emerged. It is clear that a raster only solution, though capable of providing an overwhelming amount of information, has a number of limitations due to the nature of raster files and thus the need for vector overlays to cover these limitations is more demanding.*

Today, the three methods co-exist in the domain of Web mapping and hold the lion's share of Web mapping applications available on the Web. Certainly, there are several other Web mapping efforts from scanned maps and maps in PDF files to flash-based mapping applications with the latter cases being the most popular among them. Yet, flash is considered inferior to SVG for cartographic purposes (Neumann 2002) while at the same time GIS functionalities in flash-based applications are rare.

Limitations of raster-only maps

While these solutions have managed to deliver spatial content to the users, they are heavily based on the pre-rendering of spatial information in a raster format. Such a strategy, though, bears a number of limitations. The major drawback is that such solutions serve inflexible and non-interactive content. Interaction requires that individual elements represented to be responsive (Neumann and Winter 2004). Thus, truly interactive maps have to provide the ability to interrogate every object represented. On the contrary, raster-based maps hinder applications to access directly the elements that compose the map without further communication with the server, and thus paying the price of network latency and server processing for any request relative to the content of the map.

Raster-only Web application try to overcome this problem by establishing a form of linkage among depicted entities, pixel coordinates and information *recorded in the spatial DB on the server by using raster maps which recognise pixel coordinates as meaningful and provide additional information that is related to the specific coordinates by extracting data from the server. However, no matter how accurate and complete this linkage can be, such communication is expensive both in terms of time and software resources because it faces network latency and server processing for any request relative to the content of the map. The solution to increased server processing demand is to have several copies of the IMS running to cope with the load but while this helps responsiveness it also raises the cost of building and maintaining the overall application.*

Additionally, while OGC's approach manages to fulfil its goals by providing ways to build open and interoperable Web mapping services, the actual implementation remains rather technically complicated compared to the use of APIs of public tile servers. Moreover, the need of extracting spatial data (from both raster and vector datasets) and rasterising the outcome is neither totally avoided nor improved and thus response times, server load and user's interaction with the mapping applications remain an issue.

As for tile servers, even though they can efficiently transmit raster images that allow users to smoothly pan and zoom the map, also introduce structural limitation in their APIs regarding overlaying additional information.

In the remainder of the section we will discuss specific limitations of raster-only maps. The interesting point in these examples lies behind the fact that the majority of the limitations discussed can be by-passed with the use of SVG.

Data Volume

One limitation is that current abilities of Google Maps API can handle efficiently about 100 points or about 1000 points in polylines and polygons which are considerably less than normal spatial applications require (Gibin et al. 2008). This generated three different breeds of Web mapping applications regarding the handle of increased data volume. The first type loads the whole volume of spatial data to the client browser but pay the price of time delay until the web page renders. The obvious disadvantage is that the response time of the application can grow to unacceptable levels and thus the advantage of using tile servers for serving pre-generated raster data is lost. The second type of applications firstly forces the user to apply logical segregations in the data and then create a map overlaid with as little spatial *data as possible and thus avoid severe delays in transmitting the map. In that way though, the user cannot see the whole 'picture' of data. Moreover, when the borders of the logical segregation are reached then there is need to roll back few steps in order to follow a different path in the logical segregation procedure. In any case the segregation is predetermined by the map author and the user cannot see adjacent 'logical areas' in one view. In the third type, the application is responsible to minimize the volume of the data by clustering the content which hinders users to see the true presence of spatial content.*

Indeed, in a preliminary test contacted in order to determine the usability of SVG with the mapping APIs it was realised that accessing and rendering the position of 1000 community-led organisations in London by using solely Google Maps API needed approximately 35 seconds. On the contrary, by overlaying SVG data on top of the raster tiles, the map needed approximately 5 seconds to access and render the data (Figure 1).

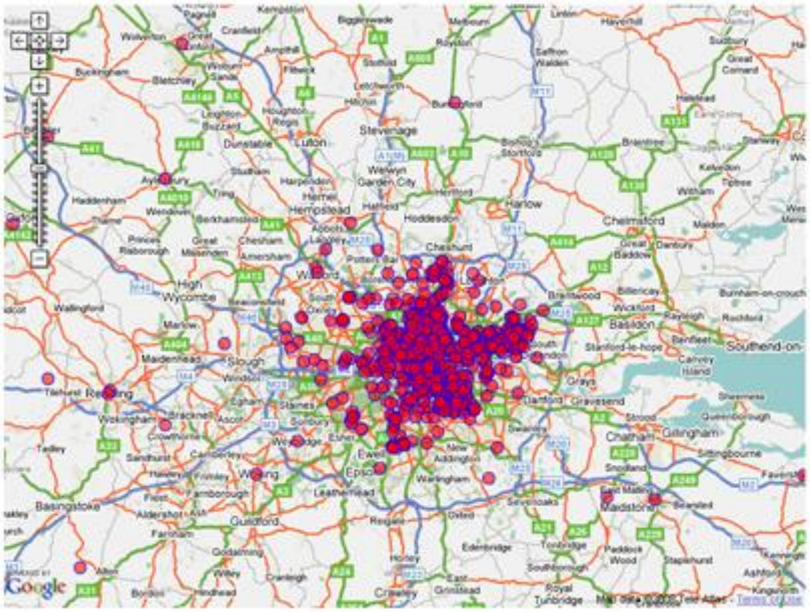


Figure 1. Points rendered using Google Maps API and SVG overlay

Cartographic inconsistencies

A second limitation is that using only raster files through map APIs, developers are imposed to overlay their datasets on top of the delivered raster tiles. Consequently, the overlaid layer (no matter what geometry or entity represents) goes on top of the topographic map obscuring some of the *content of the map (Figure 2a). This is in contrast with the cartographic principles of thematic layers and map composition where the position of each layer is dictated by a number of logical constraints including the portrayed entities' geometry (Figure 2b).*

A workaround of this problem would be APIs to allow access to individual layers of information, enabling developers to arrange thematic layers and thus compose their maps cartographically correct. Although there are efforts to tackle the problem (Gibin et al. 2008), still the issue largely remains unsolved.

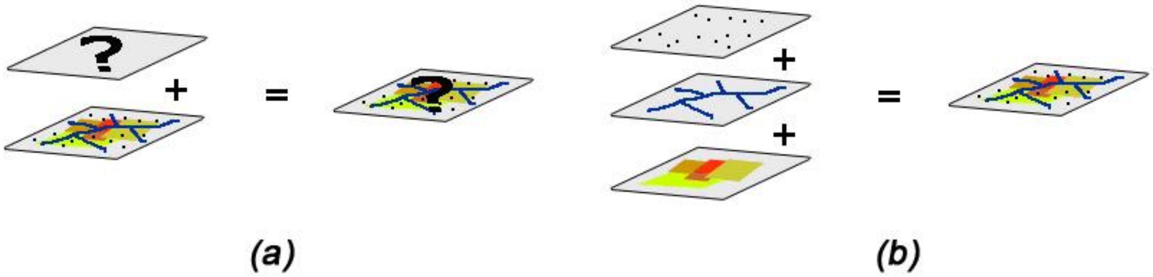
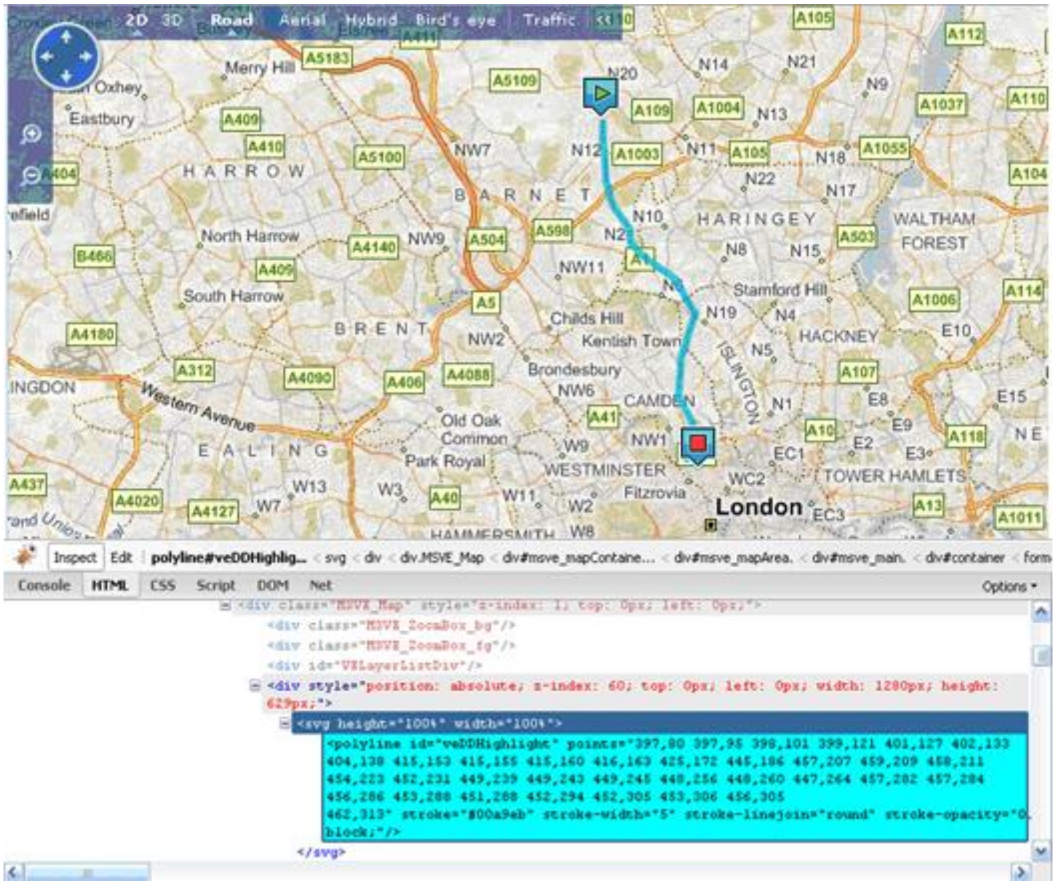


Figure 2. Map composition: (a) following API structure, (b) following cartographic principles

Limited Functionality

Another limitation, affecting all three approaches, is that the raster's static nature considerably limits the functionality that a Web mapping application can provide to the user (ESRI 2006). Thus, there is need to introduce a new layer between the map and the user in order to host the missing functionality. Major commercial map providers (Google, Microsoft, Yahoo etc.) have themselves experienced the limitations of the raster-only Web maps when it came to enhance the functionality of their mapping applications by building more task oriented functionality such as routing services (Figure 3). Such services, use primarily vector data (VML or SVG depending on the browser) in order to render the route while the use of raster tiles for the same purpose is left as a last option in case of obsolete browsers (Google).



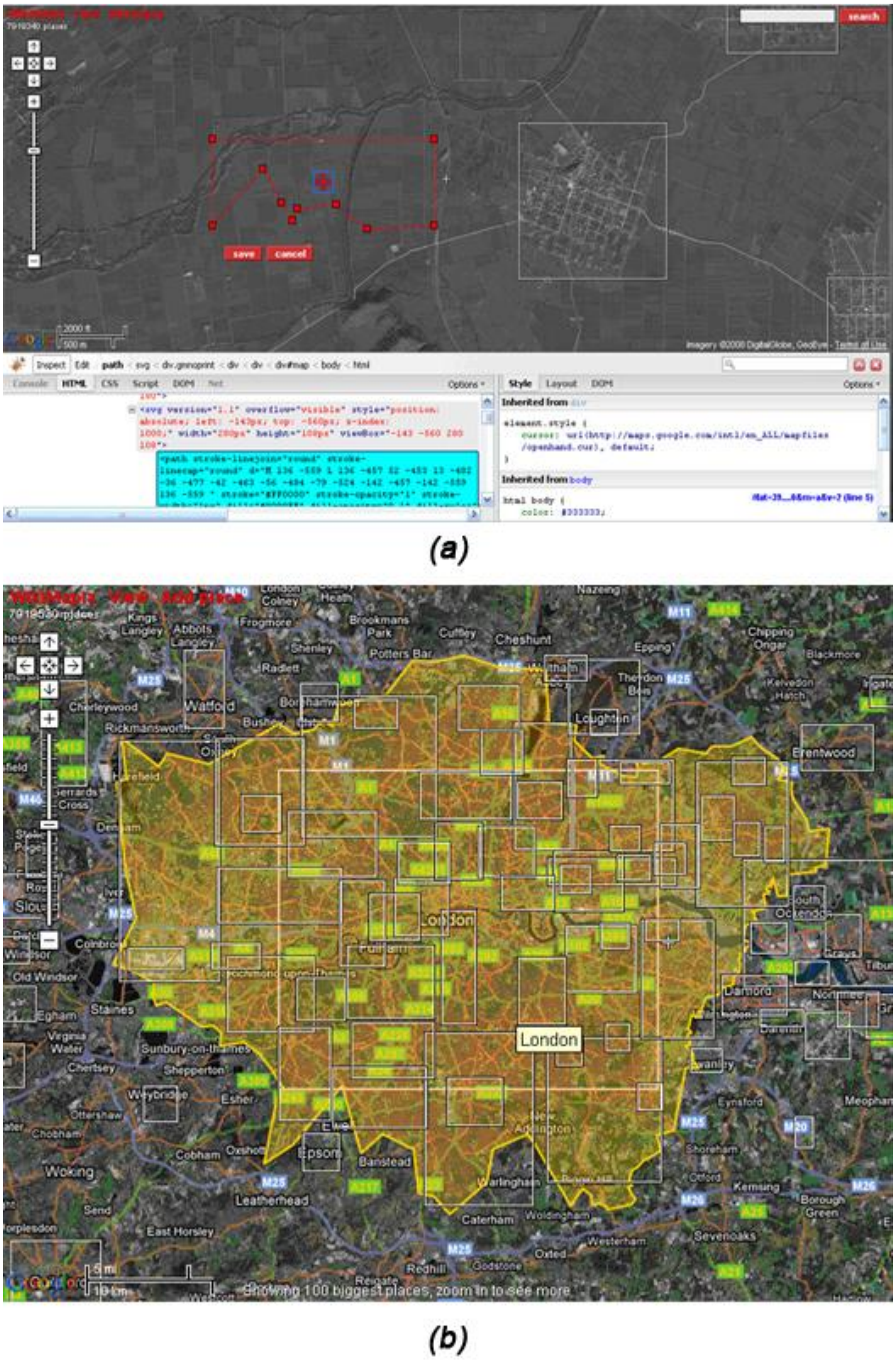
```
<div class="MSVE_Map" style="z-index: 1; top: 0px; left: 0px;">
  <div class="MSVE_ZoomBox_bg"/>
  <div class="MSVE_ZoomBox_fg"/>
  <div id="VELayerListDiv"/>
  <div style="position: absolute; z-index: 60; top: 0px; left: 0px; width: 1280px; height:
  629px;">
    <svg height="100%" width="100%">
      <polyline id="veDDHighlight" points="397,80 397,95 398,101 399,121 401,127 402,133
      404,138 415,153 415,155 415,160 416,163 425,172 445,186 457,207 459,209 458,211
      454,223 452,231 449,239 449,243 449,245 448,256 448,260 447,264 457,282 457,284
      456,286 453,288 451,288 452,294 452,305 453,306 456,305
      462,313" stroke="#00a9eb" stroke-width="5" stroke-linejoin="round" stroke-opacity="0.
      block;"/>
    </svg>
```

Figure 3. Routing Service with SVG data overlaid

Web mapping in Web 2.0

Up to now, the traditional way of creating and publishing spatial information was predominantly by expert map authors. Today, thought, this is hardly the truth. In the context of a new era in the Web, known as Web 2.0, where the main characteristics are bi-directional communication and user generated content, Web maps are mainly created by the users themselves. Researchers (Goodchild 2007, Sui 2008) discussing the phenomenon acknowledge that the power of users cannot be overlooked. An indicative example of today's Web mapping applications is wikimapia (www.wikimapia.org). Wikimapia allows users to create spatial content by digitizing spatial entities.

The inflexibility of raster-only maps cannot easily foster such actions and thus, for creating (Figure 4a) and rendering (Figure 4b) spatial content, the *application is using vector data on top of the raster backdrop map. Once again, VML and SVG are used depending on the user’s browser.*


search
save
cancel
2000 ft
500 m
Imagery ©2008 DigitalGlobe, GeoEye - Terms of Use
Inspect Edit path < svg < div.gmnoprint < div < div < div#map < body < html
Console HTML CSS Script DOM Net Options
Style Layout DOM Options
Inherited from div
Inherited from body

```
<svg version="1.1" overflow="visible" style="position:
absolute; left: -143px; top: -560px; z-index:
1000;" width="280px" height="100px" viewBox="-143 -560 280
100">
  <path stroke-linejoin="round" stroke-
  linecap="round" d="M 136 -559 L 136 -457 82 -453 13 -482
  -36 -477 -42 -463 -56 -494 -79 -524 -142 -457 -142 -559
  136 -559 " stroke="#FF0000" stroke-opacity="1" stroke-
```

```
element.style {
   cursor: url(http://maps.google.com/intl/en_ALL/mapfiles
   /openhand.cur), default;
}
```

```
html body {
   color: #333333;
```

(a)

(b)

Figure 4. Creating (a) and rendering (b) spatial content in wikimapia with the help of SVG

Specialized applications

Although public Web mapping applications hold the lion's share, still there are several cases where Web maps function as specialized tools. These tools do not aim general Web users but specific audiences that have the expertise to use efficiently such tools. In these cases, issues such as browser's native support or downloading and installing plug-ins, are not considered obstacles since the gains of using the tool easily drive users to adjust their browsers to meet the minimum specifications set by the tool. For example, the users of an application (www.polis-logic.gr/e-domisi/, beta version) that offers planning and building legislation for survey and civil engineers and uses SVG to deliver maps have proved to be more than willing to install Adobe's SVG viewer in order to have access in such specialized information on-line. Often, such applications have to offer accuracy that lacks from raster-only applications. For example, in the application mentioned earlier, on-screen measurement and entity interrogation has to allow users to visually recognize a-priori the entities involved. This can be easily achieved by adding events like "onmouseover" to vector-encoded objects but it is not trivial for raster-only maps. As Buttenfield (2002) argues, rasterised maps can introduce inaccuracies into applications that need to have access and calculate upon intrinsic vector data characteristics, such as vectors' geometry or topology.

Limitations of Vector Data

As is the case with raster-based formats, there are also intrinsic problems related to the nature of vector encoding restricting their broader adoption.

First of all, the fact that HTML specification failed to provide a friendly environment to vector-encoded data, lead to independent development of open-source and proprietary vector formats with colliding interests. In this context, and even though now all major browsers are vector-aware, today's picture of vector acceptance remains confusing: Microsoft's Internet Explorer (IE), which is the predominant Web browser, supports the proprietary VML format and in parallel the company evolves another vector based format called Silverlight. In contrast, Firefox, Safari and Opera Web browsers provide native support for SVG. Although IE is able to handle SVG files, it needs a plug-in made by Adobe. However, Adobe announced the end in evolution of *its SVG plug-in after acquiring Macromedia, the company behind Flash. In parallel, Adobe develops Flex to compete Microsoft's Silverlight. This struggle among formats makes developers extremely reluctant to invest time and money in vector based solutions.*

Another set of problems source from the nature of vector encoding. Dynamic generalisation and efficient ways to transmit vector data are still unsolved problems for cartographers, mainly due to the difficulty in holistically defining the problems in a computerised environment and the highly complicated algorithms needed for solving them. These reasons lead to unacceptable response times for a Web application even over fast communication links (Zhou and Bertolotto 2004) or poor cartographic quality.

Finally, the open nature of XML technologies that are widely used in vector encoding becomes a drawback when it comes to preserve intellectual property rights of the map data owner. Because of the XML specifications' use of plain text to encode information, which is easily accessible to humans and machines, using an XML based vector format (such as SVG and VML) means that the client's machine will have access to raw spatial information. However, there are available methods to transform real world coordinates to pixel ones before transmitting the data to the client and thus considerably hinder data theft by reverse engineering.

Weighting both limitations of raster-only and vector-only solutions, emerges that by building hybrid applications where each format is used in a way that benefits from its specific advantages is the most promising way forward.

This can be the chance that SVG waited for so long. By exploiting the impact that map APIs have and by proving that has the characteristics needed to address the limitations of a raster-only Web mapping application, SVG can assert a key role in Web mapping and Web GIS applications. Thus, the need to overcome some of the fundamental problems that follow the use of SVG is more demanding. In this context, an effort to achieve faster vector data transmission from the server to the client is presented in the next Section.

Putting AJAX into SVG grammar

Asynchronous Javascript and XML (AJAX) is a relatively new methodology to achieve client-server interaction, yet it has become very popular and has been *implemented in a variety of Web applications including Web mapping ones. For example, tile-based applications use AJAX to send only the needed tiles to the client instead of reloading the whole map and thus give the user a smooth way to interact with the map while panning. In contrast, SVG-based mapping applications use AJAX to renew the entire map area depending on users' inputs. This applies either when the SVG file is part of an html page or when the application is written only with SVG. This practice though stalls user interaction and constantly presents "please wait" or "loading" messages to the user.*

Here, a new approach in the use of AJAX is described that aims to overcome this problem and enhance user interaction. The underlying concept is to exploit the grammar of SVG while applying AJAX. When a user requests a map, the process starts by extracting tiles of geographical content from a spatial database. The content is sent to the client where some tiles are held in the <defs/> element and the rest are put in the actual map area of the SVG file (Figure 5).

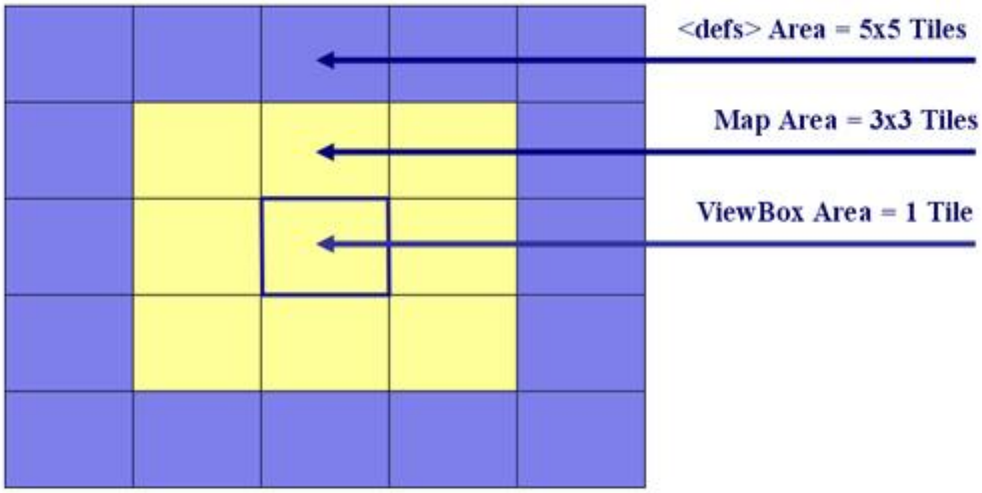


Figure 5. The allocation of tiles inside the SVG document

This approach provides a new method to transmit vector data on the client using AJAX, that eliminates lapses between user's action and map rendering but it does not solve the problem of on-the-fly vector generalization or progressive transmission of data. Thus, this approach is applicable in cases that are based in multi-resolution spatial databases. In these cases zooming *actions (in or out) request spatial content from different datasets and hence the whole procedure described earlier can start from the beginning. Interesting issues remain regarding the structure of a spatial database or database enrichment that could further facilitate data extraction for this method.*

In that way, the <defs/> element can be used as a "cache memory" for the SVG document by holding the spatial content that AJAX requests send to the client. When the user pans, and there is need to render more spatial content into the map, the application locates the additional spatial content into the <defs/> element. Then, clones the proper content into the map area and replaces the content of the out-of-view tiles of the map (Figure 6).

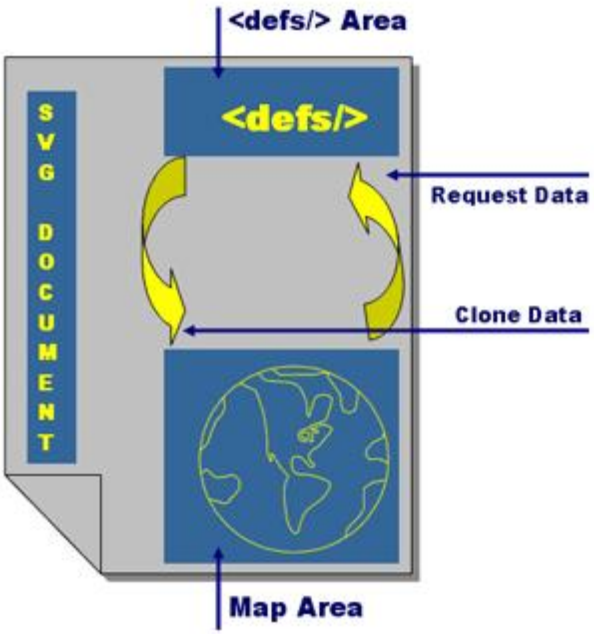


Figure 6. Requesting and cloning data inside the SVG document

In parallel, an AJAX request is triggered to add more tiles in the <defs/> area preparing, in a sense, the SVG document for the next user's action (Figure 7). That way, the user sees no delay in the rendering of the map since spatial data for rendering are constantly available while the actual AJAX request happens behind the scene without the user noticing it. In order the outcome of this procedure to have the full potentials of a vector-based map, a final step is needed. A process to reunite the tiled geometry of the depicted entities that have 1D or 2D geometry (lines and polygons) based on their unique ids should be available. In that way instead of having tiles of spatial data on the *map area, the map will be consisted of discrete spatial entities that can directly interact with the user.*

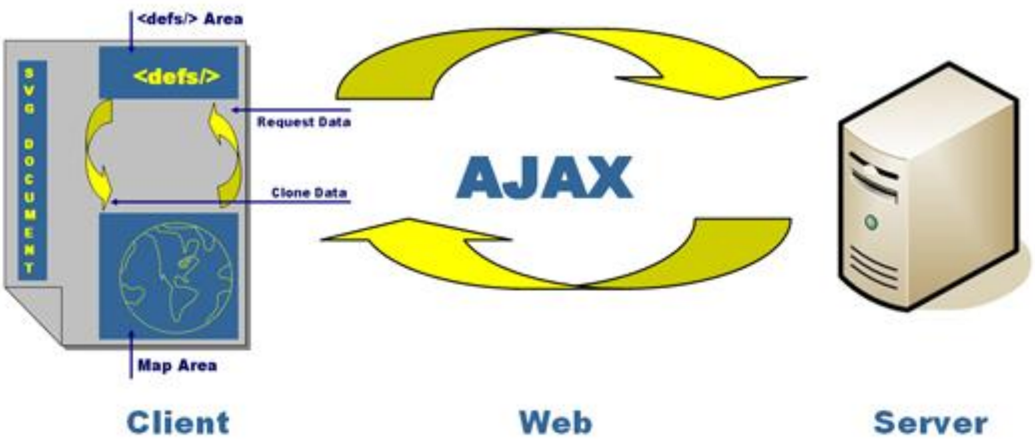


Figure 7. AJAX request from the SVG document

This approach provides a new method to transmit vector data on the client using AJAX, that eliminates lapses between user's action and map rendering but it does not solve the problem of on-the-fly vector generalization or progressive transmission of data. Thus, this approach is applicable in cases that are based in multi-resolution spatial databases. In these cases zooming actions (in or out) request spatial content from different datasets and hence the whole procedure described earlier can start from the beginning. Interesting issues remain regarding the structure of a spatial database or database enrichment that could further facilitate data extraction for this method.

Conclusion

The domains of Web mapping and Web GIS have based their evolution in the use of raster only maps exploiting technological trends in transmitting graphic information on the Web. Today, there is a huge variety of applications with different goals and target groups that are based on raster format with limited vector overlay capabilities and they manage to convey spatial knowledge. However, limitations associated with the nature of raster encoding have started to emerge and there are efforts to further promote and enhance user's experience with more informative forms of Web mapping.

On the other hand though, intrinsic problems to vectors as well as accessibility issues have to be addressed in order not to deter developers from investing time and money on vector-enabled applications. Newly introduced scripting techniques and the proliferation of increased bandwidth are promising factors that can help vectors find their way into public mapping. Moreover, vector data is needed to provide maps with flexible, interactive and semantic content with low cost in terms of response time and resources and thus play a key role in building the next generation of Web maps.

A hybrid approach which uses only the strong points of the raster methods, overlaid with vector data provides the means for spatial entities to host directly scripting, animation and attribute data allowing instant interaction between the user and the map. This proves to be the most efficient way to deliver spatial data even for complex Web mapping applications.

In this context, an innovative way to transmit vector data exploiting SVG grammar has been presented here. This method gives the user a smooth way to interact with the map while panning, similar with that introduced from tile servers and raster data.

Finally, answering the question posed at the beginning, by examining both theory and practice we can conclude that SVG is truly needed in order to build the next generation of Web maps.

Bibliography

Buttenfield, B.P., 2002. Transmitting vector geospatial data across the Internet. In Geographic Information Science. Second International Conference GIScience 2002, M. Egenhofer and D. Mark (Eds.), Lecture Notes in Computer Science Vol. 2478 (Berlin: Springer), pp. 51–64.

ESRI, 2006. Comparing Vector and Raster Mapping for Internet Applications, An ESRI White Paper, August 2006. Available at www.esri.com/library/whitepapers/pdfs/vector-raster-mapping.pdf.

Gibin, M., Singleton, A., Milton, R., Mateos, P., Longley, P., 2008. Collaborative mapping of London using Google Maps: The LondonProfiler. CASA working paper. Available at http://www.casa.ucl.ac.uk/working_papers/paper132.pdf. Last accessed 08 July 2008

Goodchild, M., F., 2007. Citizens as voluntary sensors: Spatial data infrastructure in the world of Web 2.0. International Journal of Spatial Data Infrastructures Research, 2, 24–32

Google, 2008. Map Overlays. Available at http://code.google.com/apis/maps/documentation/overlays.html. Last accessed 30 May 2008

Neumann, A., 2002. Comparing .SWF (Shockwave Flash) and .svg (Scalable Vector Graphics) file format specifications. Available at http://www.carto.net/papers/svg/comparison_flash_svg/index07.shtml. Last accessed 30 Jan 2008.

Neumann, A., Winter, A.M., 2004. Vector-based Web Cartography: Enabler SVG. Available at http://www.carto.net/papers/svg/index_e.shtml. Last accessed 01 Feb 2008.

Peng, Z.R., Tsou, M.H., 2003. Internet GIS: Distributed geographic information services for the Internet and wireless networks. Wiley, Hoboken, NJ. 679 pages.

Plewe B., 1997. GIS Online: Information Retrieval, Mapping, and the Internet. Santa Fe, NM, OnWord Press. 311 pages.

Sui, Z., D., 2008. The wikification of GIS and its consequences: Or Angelina Jolie’s new tattoo and the future of GIS. In Computers, Environment and Urban Systems 32 (2008) 1–5.

Zhou, M., Bertolotto M., 2004A Data Structure for Efficient Transmission of Generalised Vector Maps. In M. Bubak et al. (Eds.): ICCS 2004, LNCS 3039, pp. 948–955, Springer-Verlag Berlin Heidelberg 2004.