# ISIG/WEB

## *A web-based tool for viewing and editing maps*

Gustavo Henrique Mandolesi, Cláudia de Andrade Tambascia, Sandro Danilo Gatto and Alexandre Melo Braga

*CPqD Telecom & IT Solutions - Rodovia Campinas-Mogi Mirim km 118,5 - 13086-902, Campinas, São Paulo, Brazil*

Abstract: Free Software has been widely discussed by Brazilian private organizations as well as Federal Government. Brazilian politicians and economists are interested in the use of Free Software platforms for social services provisioning. Not only because this kind of product can lead to important cost savings, but also because it can become a key point to the nation interests, in terms of internal development and competition regarding foreign markets. This paper presents a Web-based distributed platform for visualization, manipulation and edition of maps, which can potentially improve the government efficiency in planning, provisioning and management of social actions.

Key words: GIS; free software, scalable vector graphics, java, J2EE, Open GIS Consortium.

## 1. INTRODUCTION

Geographic Information Systems (GIS) are computer systems built specially to tackle geographic information problems, such as storage, analysis and manipulation of geographic coordinates and related data. This kind of data can represent both physical objects and environmental phenomena that have geographic location as inherent characteristic. Data is collected from several different sources and usually stored into geographic database management systems, which are indispensable components for any GIS.

One of the main characteristics of GIS platforms is their ability to gather information from various sources. Another remarkable feature of these systems is the possibility of presenting data to the final user in ways that facilitate not only the visualization but also the edition and overall manipulation of geographic data.

Brazilian Government intends to promote the adoption of Free Software by public administration. Many efforts have been done and actions have been accomplished in this direction. The platform presented in this paper is one of these efforts.

Generally speaking, Free Software is a computer program in which the traditional copyright rules are relaxed, in a way that access to the source code is granted, with no restrictions, for copy, distribution and modification.

ISIG/WEB stands for "Inovação em Sistemas de Informação Geográfica na Web" (Web GIS Innovation) and is a large project funded by FUNTTEL (Fundo para Desenvolvimento Tecnológico das Telecomunicações) aiming software development to attend Government needs regarding GIS applications.

## 2. WEB/GIS INTEGRATION

As GIS applications have become important, the issue of spreading this kind of information in a wider scale has arisen. At the same time, the Internet has become very popular, evolving as an appropriated channel for geoinformation distribution, and the integration of these two environments has been a natural consequence.

This kind of Internet application has special requirements, mainly security and platform independence issues.

One of the strategies to make Internet GIS available is based on the classical client-server model, through requests and services exchanges. Different implementation types exist, some concentrating geographic processing on the server side, some on the client side, and some using hybrid architectures, where processes are combined depending on specific requirements.

Usually, server-side GIS present all the functionalities to handle user demands via web browser. In this case, the server is many times overloaded, requiring a great amount of storage and processing capabilities to satisfy user requests. On the other hand, client-side GIS lead to performance impacts on the client machine after data has been transferred to it

The decision of which strategy to use in the project of a GIS application has to take into account the characteristics and functionalities of the required system, once both cases present advantages and disadvantages that have to be balanced according to the application.

# 3. THE ISIG/WEB PROJECT

The ISIG/WEB project was developed by CPqD[1] and is a prototype product based on a distributed GIS architecture that allows visualization and edition of geographic information on Web-based maps. Version 1.0 is composed of three main modules: the Map Editor, the Style Editor and the Document Transformation Module.

The product environment is composed of: a map server; a feature server; a database server to store spatial data; a Web application server to handle HTTP requests; and clients, represented by browsers.

The Internet-based visualization and edition of geographic data is possible due to the functional implementation of Web Map Service (WMS) and Web Feature Service (WFS) protocols, open standards specified by the OpenGIS Consortium[2]. These services are available through an application server that handles WMS and WFS requests, delivering geographic data stored in a database server.

## 3.1 System modules

Figure 1 shows the composition of ISIG/WEB in terms of its modules.

The Map Editor is a module that allows visualization of maps in Scalable Vector Graphics (SVG) format and edition of spatial and scalar data. The map visualization is based on WMS 1.1.1 standard requests to a map server. Spatial and scalar edition are achieved through WFS 1.0.0 compliant requests dispatched to a feature server.

The Style Editor is used to graphically edit map objects symbols using the OpenGIS Styled Layer Descriptor (SLD) 1.0.2 standard. The map server uses style information to render map images.

The Transformation Module is the main component behind map server functionality. It processes the map requests sent by the Map Editor. According to the map requested by the client, the Transformation Module makes feature requests to the feature server and analyses the corresponding SLD document. This document contains information on how features should be drawn on the response map.

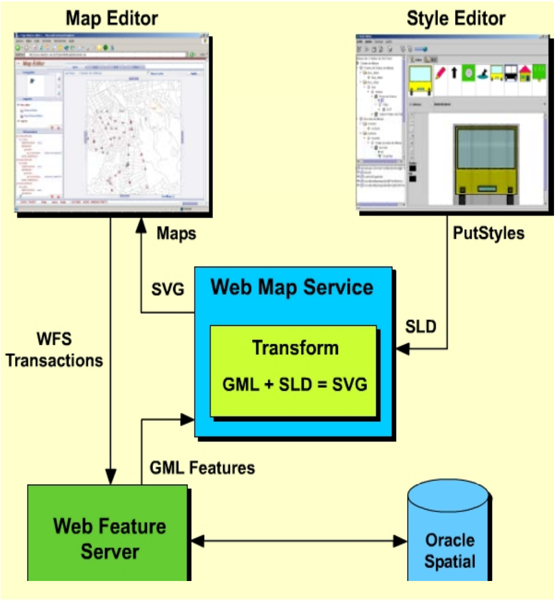


*Figure 1.* Structure of ISIG/WEB modules

The ISIG/WEB version presented in this paper has used the Deegree Project[3] WMS and WFS demo implementations[4] to test the application. However, by the time the system was developed, DeegreeWFS didn't fully covered WFS 1.0.0 specification, and this fact has led to some difficulties when publishing and editing data into Oracle Spatial, the geographic database server used in the project. Consequently, some modifications had to be included into its source code in order to adjust it to the project requirements.

## 3.2 Architecture

The system architecture is divided into functional layers that allow the division between application components and third-party components. This structure brings benefits in terms of easier maintenance and reuse, and it's shown in Figure 2.

The client layer includes the application business modules and external client applications, like Web browsers and virtual machines. Sub layers responsible for presentation, business rules and access to data from Style Editor and Transformation Module compose it.

That layer uses the Virtual Platform layer to obtain access to middleware resources and the Business layer to get support from subsystems.

The Virtual Platform defines an architectural model that isolates the Business layer components from the middleware components. This way, the application development can be made in a transparent manner, independently from products like database servers and application servers. This kind of transparence is achieved due to an API that can be contract-based or message-based.

The Middleware layer supplies support services to the application business, and is represented by the map server. The project role is to configure and make use of middleware application support services.

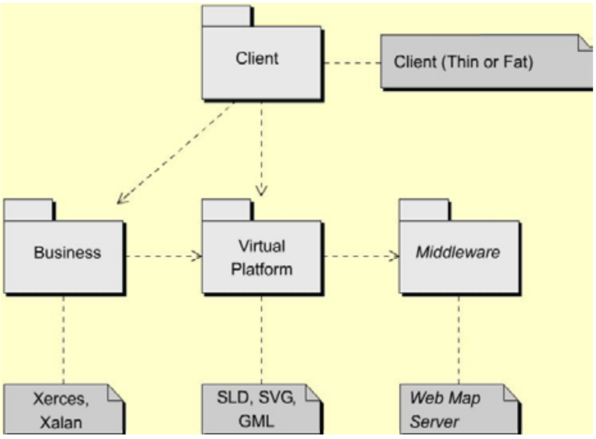


*Figure 2.* ISIG/WEB components architecture

The Business layer has generic business logic classes. Usually, they compose subsystems or frameworks that can be extended or reused somehow in the project.  The system uses some components from other projects like Xalan[5] and Xerces[8], which support some business rules related to the application layer.

From a logical perspective, the application is divided into three different layers. The Presentation layer is composed of Java Swing components (desktop modules) and JSP pages (for Web modules). Users interact with the system through these components, which represent the "View" on the Model-View-Controller (MVC) Design Pattern, providing content presentation and user interaction.

The Business Rules layer is composed of Java classes related to application domain and implements Model (information modeling) and Controller (transactions managing and logical flow) of MVC design pattern.

## 3.3 Map Editor

The Map Editor is a Web application that presents a graphic interface accessible to clients through the Internet and has a server component responsible for receiving and responding requests.

The graphic interface makes it possible to the user to view maps composed of multiple features, based on SLD and GML schemas. Maps are SVG documents and the Adobe SVG Viewer plug-in is used for visualization in Microsoft Internet Explorer and in Mozilla Firefox browsers.

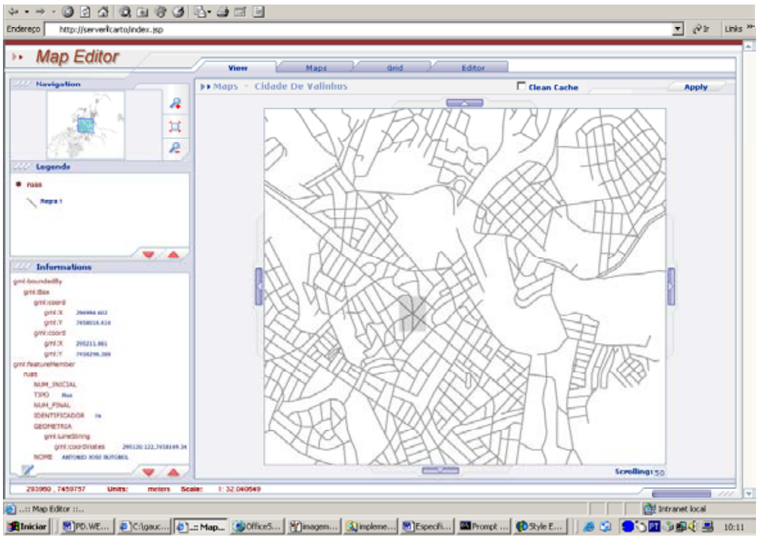


*Figure 3.* ISIG/WEB map viewer

Using the Map Editor it is also possible to manage map features, editing, inserting and removing features from the feature server.

The Viewer presents several visualization styles, based on SLD schemas, on which map layers are composed. For any specific layer, the user may define different styles. Figure 3 presents the interface the user interacts with to use Viewer Functionalities.

The Web interface is divided into functional tabbed panes, comprehending four working areas: Vision, Maps, Grid and Editor.

Vision is the Map Editor main area, where functions related to map visualization are available. Within Maps, it is possible to define categories and layers that the user can select for visualization. The Grid is responsible for scale and map fragmentation level definition, and exhibition of quadrants loaded by the user. Finally, the Editor manages map features, and allows feature retrieval based on map, layer or feature type selection.

Map Editor main functionality is to load and present map images. By the time the user clicks on "Maps" pane, the Editor dispatches a GetCapabilities request to the WMS server, and is supposed to receive a XML document that contains a description of all available layers, it's presentation order and all possible operations. Based on this information, it's possible to configure and sort layers for visualization.

Then, the user can select desired category layers, what generates a GetMap request sent to the WMS server, which returns a SVG document containing the required map image.

The user is able to customize some aspects of the visualization interface. He has the possibility to define which map information will be displayed, select data presented in the legend, configure map visualization scale and area. These customization features allow the user to directly define georeferenced data to be displayed and the way they will be presented.

A plug-in has to be used for SVG visualization in the browser. Its advantages are:

- Great compatibility with existing platforms;
- It makes the application simpler, since the plug-in API supplies all object manipulation functionality (and not the application itself);
- It has good performance concerning execution time.

## 3.4 Style Editor

The Style Editor enables the user to visualize and edit different kinds of map styles, symbols, legends, associations, layers and features. The main idea behind it is to provide easy visual map composition based on a GML schema.

The Editor is a Java desktop client application, built upon multi-layered client-server architecture. Its distribution and execution uses Java WebStart technology, what makes the application deployment process automatic and simple. By clicking on a link in his workspace or in a Web page, the user can start on his own machine the most recent version of the desktop application, available somewhere in the network. This way, it offers the interaction possibility, sophistication and ease of use of any desktop application, combined with distribution capabilities provided by the Web.

Styles are described in XML format, compliant to OGC Styled Layer Description specification. They are the basic unit for SVG generation and visualization into Map Editor.

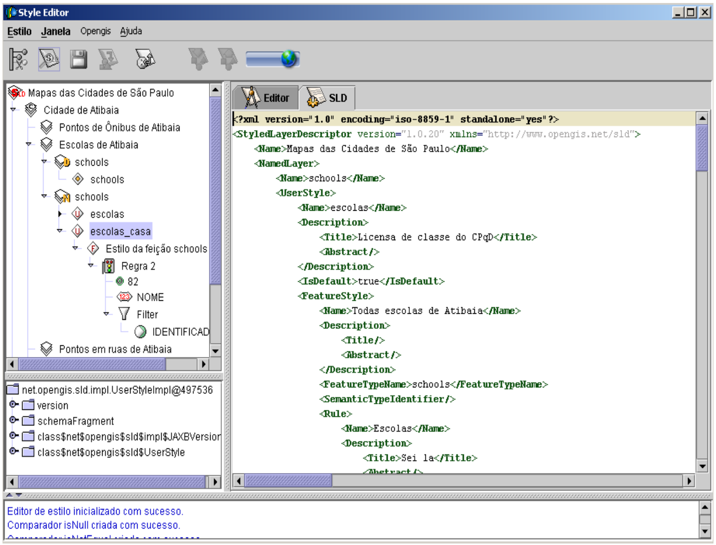
Figure 4 presents the Style Editor graphic interface, and shows the menu for loading layer names through the GetCapabilities WMS request. Right below is the layer tree containing the layers available in the server.

The Style Editor uses the SLD schema to create a hierarchic structure of information through the layer tree. For each element, there is an associated editor that contains information related to the element being edited.

There's also the possibility of visualizing SLD information corresponding to the style being edited in XML format, just like it's stored in the server. This is achieved by activating the SLD pane located in the graphic interface central panel.

Regarding OpenGIS standards compatibility, the Style Editor has operations that handle some of WMS, WFS and SLD specifications.

The Editor generates visualization styles related to map layers. For a specific layer the user can define several different styles. However, the component assumes that there's a WMS server integrated with a WFS server that supplies information about available layers and the kind of features that compose them.



```
<?xml version="1.0" encoding="iso-8859-1" standalone="yes"?>
<StyledLayerDescriptor version="1.0.20" xmlns="http://www.opengis.net/sld">
    <Name>Mapas das Cidades de São Paulo</Name>
    <NamedLayer>
        <Name>schools</Name>
        <UserStyle>
            <Name>escolas</Name>
            <Description>
                <Title>Licensa de classe do CPqD</Title>
                <Abstract/>
            </Description>
            <IsDefault>true</IsDefault>
            <FeatureStyle>
                <Name>Todas escolas de Atibaia</Name>
                <Description>
                    <Title/>
                    <Abstract/>
                </Description>
                <FeatureTypeName>schools</FeatureTypeName>
                <SemanticTypeIdentifier/>
                <Rule>
                    <Name>Escolas</Name>
                    <Description>
                        <Title>Sei la</Title>
                        <Abstract/>
```

```
Editor de estilo inicializado com sucesso.
Comparador isNull criada com sucesso.
```

*Figure 4.* ISIG/WEB style editor

The following requests are handled by OGC services implemented by the Style Editor:

- GetCapabilities: WMS service to retrieve information about available layers in the server;
- GetMap: WMS service that returns a SVG map, according to Map Editor user configurations;

- GetLegendGraphic: this service supplies information relative to objects of viewed layers;
- GetFeatureInfo: WMS service that returns information about features of any specific map position, selected by the user for visualization;
- GetStyles: returns user defined styles;
- PutStyles: this service is used to store new user defined styles. This way, many styles can be defined and stored for different layers, becoming available for future requests;
- DescribeLayer: returns WMS information about features used in a specific layer;
- DescribeFeatureType: WFS information about a specific geographic feature;
- TransactionDelete, TransactionInsert and TransactionUpdate: WFS services for removing, inserting and updating map features;
- DeleteSymbol, InsertSymbol, UpdateSymbol, and GetSymbol: these services allow removing, inserting and updating symbols in the symbol library;
- GetAllSymbols: returns all symbols from the symbol library;
- GetLibrary: returns the symbol library.

Figure 5 shows an architectural view of Style Editor, presenting defined OGC services.

The Style Editor accesses a symbol library to graphically represent geometric attributes of geographic features. It requests the available symbology to a symbol library manager. As an answer, it receives a list of SVG graphics that represent the symbols that may compose a visualization style.

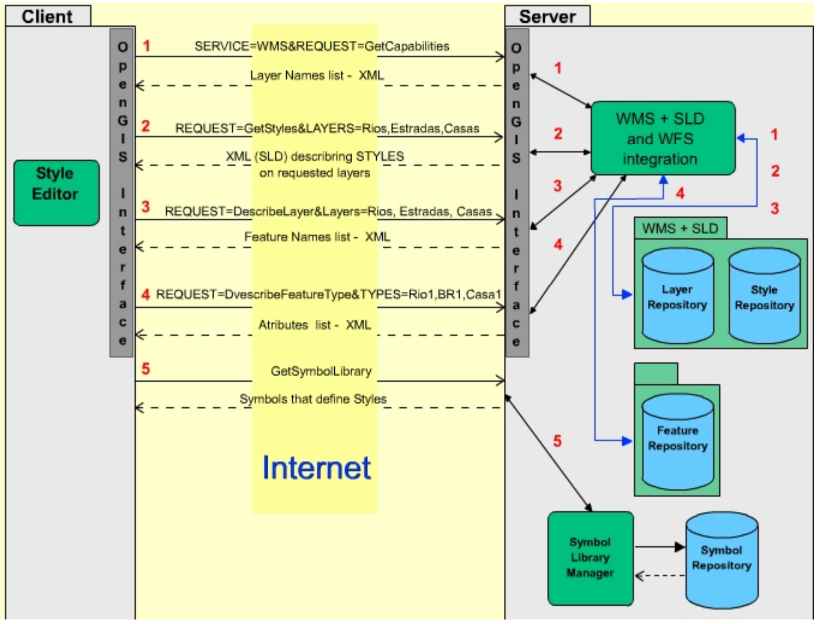

Client
Server
OpenGIS Interface
Style Editor
1 SERVICE=WMS&REQUEST=GetCapabilities
Layer Names list - XML
2 REQUEST=GetStyles&LAYERS=Rios,Estradas,Casas
XML (SLD) describring STYLES on requested layers
3 REQUEST=DescribeLayer&Layers=Rios, Estradas, Casas
Feature Names list - XML
4 REQUEST=DvescribeFeatureType&TYPES=Rio1,BR1,Casa1
Atributes list - XML
5 GetSymbolLibrary
Symbols that define Styles
Internet
OpenGIS Interface
1
2
3
4
5
WMS + SLD and WFS integration
1
2
3
4
WMS + SLD
Layer Repository
Style Repository
Feature Repository
Symbol Library Manager
Symbol Repository

*Figure 5.* Interaction between Style Editor and client

## 3.5 Transformation Module

SLD files are combined with GML documents to generate SVG images. However, SLD is not a format that can be interpreted directly by a transformation engine. So, it's necessary to use a declarative XSLT format to collaborate with the transformation.

The rules for converting from SLD+GML into SVG depend directly on the XML schema for the SLD language defined in the OpenGIS specification, and this file has to be modified if the SLD specification changes.

The OpenGIS offers the freedom to implement fat or thin clients for map visualization, as presented in Figure 6 and Figure 7. Fat clients require less functionality from the server and from other components. Thin clients are easier to develop and embedded into general-purpose software components.

Thin clients are simple software units with limited functions and flexibility, which require less CPU and memory. They are usually indicated for handhelds and mobile devices.

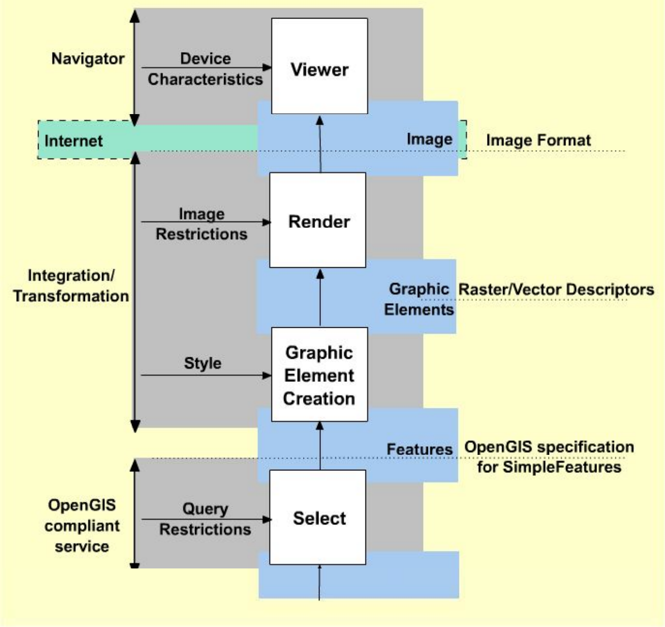


*Figure 6.* OGC services thin client

Fat clients generally require a significant amount of computer resources, but have great flexibility and capacity of decoding, transforming, building and interacting with stored geographic data.

For thin clients, the Transformation Module must be on the server side. Generally it's used when there are raster images (for example GIF or JPEG images transferred to the client) or vector images (in SVG format).

However, for a fat client, the Transformation Module would be located in the client itself, which would receive GML data and transform it to the appropriate visualization format.

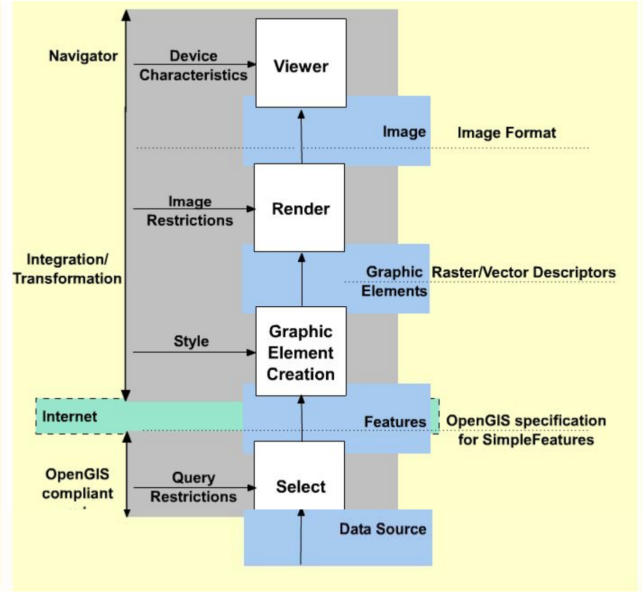


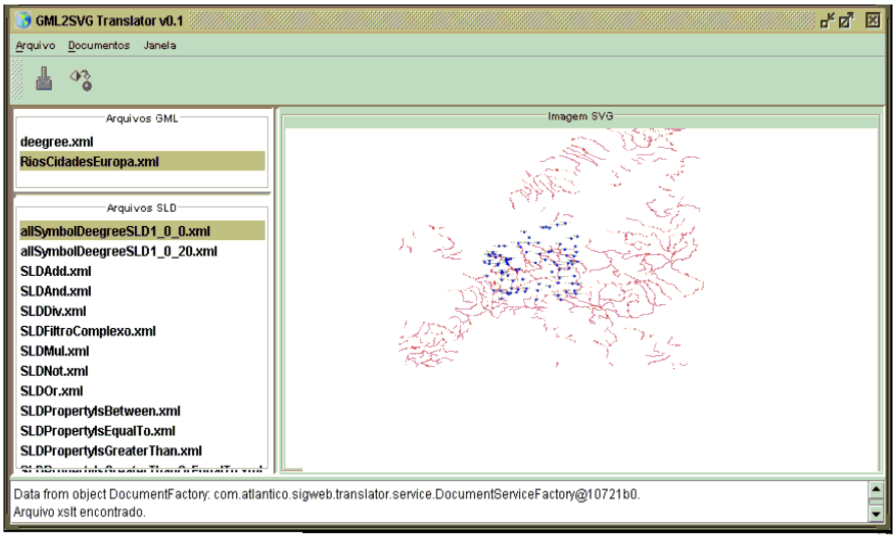
*Figure 7.* OGC services fat client

The Map Editor is a thin client and the transformation occurs at server-side, using XSLT. The SVG document generated from SLD and GML is then sent to the client browser.

Besides, there is also a desktop implementation of the Transformation Module, presented in a Java Swing interface. In this case, as it is a fat client, the transformation is made locally.

Figure 8 demonstrates the desktop version of the Transformation Module. The Module is independent of graphical interface, and was projected to serve as resource to the OpenGIS server in the Visualization Module. This graphical interface was developed to facilitate tests during the project development - the transformation itself depends on a set of documents XSLT that can be manipulated by different conversion frameworks.

Independently of the used version, the Transforming Module is fundamental, so that archives can be manipulated by other ISIG/WEB components.



deegree.xml
RiosCidadesEuropa.xml

allSymbolDeegreeSLD1_0_0.xml
allSymbolDeegreeSLD1_0_20.xml
SLDAdd.xml
SLDAnd.xml
SLDDiv.xml
SLDFiltroComplexo.xml
SLDMul.xml
SLDNot.xml
SLDOr.xml
SLDPropertyIsBetween.xml
SLDPropertyIsEqualTo.xml
SLDPropertyIsGreaterThan.xml

```
Data from object DocumentFactory: com.atlantico.sigweb.translator.service.DocumentServiceFactory@10721b0.
Arquivo xslt encontrado.
```

*Figure 8.* Desktop version of Transformer

# 4. ACHIEVED RESULTS

The ISG/WEB project was planned and developed to be an OGC compliant architecture, implemented under total compatibility with its specifications.

For this reason, it can be easily modified in order to allow the coupling of third-party components and future improvements.

Access to spatial data can be achieved using different database servers, such as Oracle Spatial, PostGIS or MySQL, for example. The system can be easily configured to access any of these databases. It is only necessary to change a XML configuration file from the WMS server.

As ISIG/WEB presents a totally modular and framework-based architecture, it is perfectly viable to reuse the system as a whole or only some of its modules.

The implementation of the system through open languages and standards (Java, Javascript and XML) favors the integration of ISIG/WEB with other systems and technologies.

The use of Design Patterns in the development of the application demonstrates an investment in product quality, facilitates the understanding of the system, and allows new functionalities to be inserted in existing software with a minimum impact.

ISIG/WEB is strongly based on some elements that constitute part of the Java 2 Enterprise Edition (J2EE) architecture, the most currently used standard for development of corporative applications and distributed systems.

At the moment, the Map Editor Module offers a functional implementation for browsers that have the Adobe SVG plug-in, i.e., Mozilla Firefox and Microsoft Internet Explorer (which is not free software).

It is recommended to test ISIG/WEB implementations with newer versions of Deegree servers, in order to adapt to evolutions made to the framework.

Currently, the only supported graphical format for map visualization is Scalable Vector Graphics. However, for clients that do not support visualization of this type of images, there is the possibility of implementing a component to transform SVG documents into raster images.

The ISIG/WEB Project contains documentation produced contemplating reports and user manuals, and detailing the stages of analysis and project. These technical documents describe classes and components that compose the system, using the Unified Modeling Language (UML).

## 5. CONCLUSIONS

This paper's intent was to present an overview of the ISIG/WEB project, a GIS Web system for visualization and edition of geographic information, based on services specified by the OpenGIS Consortium.

As a general recommendation, the use of the ISIG/WEB Project shall be considered in the development of a client for visualization and edition of maps through the Web, as well as the use of Deegree WMS and WFS server for implementation of OpenGIS services.

## 6. REFERENCES

1. CPqD - Centro de Pesquisa e Desenvolvimento em Telecomunicações. Available at: http://www.cpqd.com.br.
2. OPEN GIS CONSORTIUM. OGC Home Page. Available at http://www.opengis.org/. Accessed on: 07 mar. 04
3. DEEGREE. Deegree Home Page. Available at: http://Deegree.sourceforge.net/. Accessed on: 15 mar. 04

4. DEEGREE. Deegree Demo Site. Available at: http://demo.Deegree.org/. Accessed on: 15 mar. 04
5. APACHE SOFTWARE FOUNDATION. Xalan-Java version 2.6.0. Available at: http://xml.apache.org/xalan-j. Accessed on March, 21st, 2004
6. WORLD WIDE WEB CONSORTIUM. Extensible Markup Language (XML). Available at: http://www.w3c.org/XML. Accessed on: March, 20th, 2004
7. OPEN GIS CONSORTIUM. OpenGIS Abstract Specification: The Coverage Type and its Subtypes. Available at: http://www.opengis.org/docs/00-106.pdf. Accessed on: March 20th, 2004
8. THE APACHE XML PROJECT. Available at: http://xml.apache.org/. Accessed on March 20th, 2004