A WEB SERVICE APPROACH TO GEOGRAPHICAL DATA DISTRIBUTION AMONG PUBLIC ADMINISTRATIONS

L. VACCARI[1], A. IVANYUCKOVICH[2] AND M. MARCHESE[2]

[1]*Provincia Autonoma di Trento, Trento, Italy*
[2]*Department of Information and Communication Technology, University of Trento, Trento, Italy*

Abstract: In this paper a service-oriented architecture (SOA) is proposed to support the interaction with legacy Geographical Information Systems (GIS) and the implementation of value added data sharing services. In particular, we base our proposed architecture both on the standardization effort carried out by the Open Geospatial Consortium (OGC) and on current state-of-the-art Web Service middleware infrastructure. We have evaluated the proposed architecture in the context of GIS application integration in a departmental back-office scenario. The advantages of a service-oriented architecture are twofold: on one hand, it is possible to integrate several GIS application and data sources simply by wrapping their (legacy) services with appropriate interface and registering them in Web Service directories; on the other hand, this new service paradigm can be used to support the creation of completely new cartographic data sharing services.

Key words: e-Government, Geographical Information Systems, Service Oriented Architectures, Distributed Information Systems

1. INTRODUCTION

In recent years, the domain of geographic information has experienced a rapid growth of both computational power and quantity of information. Moreover, there is an increasing necessity to share geographic information

between different stakeholders (departments in public administration, professionals, citizens, etc) and diverse information systems in order to enable its coherent and contextual usage. This necessity is at the basis of a number of international and national projects, among which: (1) INSPIRE [1] that list among its main objectives: "*geographical data shall be made available for access and view free of charge by citizen and other users, with delivery, downloading and re-use on harmonized terms and conditions*; (2) the Italian "LABSITA", "Centro Interregionale" and "Intesa Stato Regioni" projects [2], focused on the issue of interoperability among existing geographical databases and related administrative procedures managed by local administrations. Furthermore, at the local level, there are specific projects that have to be coordinated with these higher level projects: for example the internal publication of the geographical data and metadata, the support to the formal exchange of the data with other public administrations within intra-departmental administrative procedures (like the Environmental Evaluation Procedure – "VIA: Valutazione di Impatto Ambientale"). It is important to reach these objectives using the most innovative technological framework and software architectures available at present and integrating them into the overall framework developed in European and national projects.

In this paper, we propose a service-oriented architecture (SOA) to support the interaction with legacy Geographical Information Systems (GIS) and the implementation of value added data sharing services. In particular, we base our proposed architecture both on the standardization effort carried out by the Open Geospatial Consortium (OGC) [3] and on current state-of-the-art Web Service middleware infrastructure. We have evaluated the proposed architecture in the context of GIS application integration in a departmental back-office scenario. The advantages of a service-oriented architecture are twofold: on one hand, it is possible to integrate several GIS applications and data sources simply by wrapping their (legacy) services with appropriate interface; on the other hand, this new service paradigm can be used to support the creation of completely new cartographic data sharing services.

The remainder of this extended abstract is organized as follows. In Section 2 we review current OGC specification addressing the GIS interoperability problem. In Section 3 we review the Service-Oriented Architectural model. In Section 4 we sketch the functionalities of the integrated GIS applications based on SOA. The discussion of results and related and future work concludes the paper.

2. OPEN GEOSPATIAL CONSORTIUM: WMS AND WFS SPECIFICATIONS

The Open Geospatial Consortium [3] has proposed specific and detailed specifications, for the interoperability of the geographical databases that are independent from the Web application technology used in the presentation layer. The basic idea is that an increasing number of organizations will offer their geo-referenced data according to these specifications. As specifications will become a de-facto or de-jure standard, an user application will be able to request data from different geographical service providers. The advantage when using standards is that it will be easier to combine data from different suppliers. The user will be able to request specific data and customize his data to perform personalized analysis.

At present OGC is supporting a number of standard specifications. In the present work we focused on two main specifications, namely Web Map Service (WMS) [4] and Web Feature Service (WFS) [5];

- WMS can be used to produce maps of spatially referenced data dynamically from geographic information. This specification is also an International Standard and defines a "map" to be a representation of geographic information as a digital image file suitable for display on a computer screen. WMS-based maps are generally rendered in a pictorial format such as PNG, GIF or JPEG, or occasionally as vector-based graphical elements in Scalable Vector Graphics (SVG) or Web Computer Graphics Metafile (WebCGM) formats. The WMS allows a client to overlay map images for display served from multiple Web Map Services on the Internet.
- In a similar fashion, WFS allows a client to retrieve geospatial data encoded in Geography Markup Language (GML) from multiple Web Feature Services.
- WMS and WFS operations can be invoked using a standard web browser by submitting requests in the form of Uniform Resource Locators (URLs).

A server that implements the WMS specification has to support two mandatory operations (*GetCapabilities* and *GetMap*) and can support one optional operation (*GetFeatureInfo*).

- The purpose of the mandatory *GetCapabilities* operation is to obtain service metadata (an XML document), which is a machine-readable (and human-readable) description of the server's information content and acceptable request parameter values. The client can use the results of this operation to formulate the next request. Moreover, it can build a catalog useful for the user that can choose the desired geographical layer.

- The *GetMap* operation returns a map. Upon receiving a *GetMap* request, a WMS either satisfies the request or issues a service exception. The client has to send the parameters to specify, for example, the number and the name of the layers, the bounding box to be show, the projection and the coordinate system, the raster format, the display size and so on. Several layers can be picked from different servers and can be used to make a single map. The response to a valid *GetMap* request is a map of the spatially referenced information layer requested, in the desired style, and having the specified coordinate reference system, bounding box, size, format and transparency.
- *GetFeatureInfo* is an optional operation. The *GetFeatureInfo* operation is designed to provide clients of a WMS with more information about features in the pictures of maps that were returned by previous Map requests. The canonical use case for *GetFeatureInfo* is that a user sees the response of a Map request and chooses a point (I,J) on that map for which to obtain more information. The basic operation provides the ability for a client to specify which pixel is being asked about, which layer(s) should be investigated, and what format the information should be returned in. Since the WMS protocol is stateless, the *GetFeatureInfo* request indicates to the WMS what map the user is viewing by including most of the original *GetMap* request parameters (all but VERSION and REQUEST). From the spatial context information (BBOX, CRS, WIDTH, HEIGHT) in that *GetMap* request, along with the I,J position the user chose, the WMS can (possibly) return additional information about that position.
  The actual semantics of how a WMS decides what to return more information about, or what exactly to return, are left up to the WMS provider.

A server that implements the OGC WFS specification can distribute geographic features to a client application. Moreover the WFS offers the possibility to the users to load vector data only for requested layers. The state of a geographic feature is described by a set of properties where each property can be thought of as a tuple *(name, type, value)*, following [6,7]. Geographic features are those that may have at least one property that is geometry-valued. The geometries of geographic features are restricted to simple geometries, i.e. geometries for which coordinates are defined in two dimensions and the delineation of a curve is subject to linear interpolation. The traditional 0, 1 and 2-dimensional geometries defined in a 2-dimensional spatial reference system are represented by points, line strings and polygons. In addition, the OGC geometry model allows for geometries that are collections of other geometries - either homogeneous multi-point, multi-line string, and multi-polygon collections or heterogeneous geometry collections.

Finally, GML allows features that have complex or aggregate non-geometric properties.

WFS is an interfaces for describing data manipulation operations (like Create/Delete/Update/Get feature instances) on geographic features using HTTP as the distributed computing platform. In particular, a WFS request consists of a description of query or data transformation operations that are to be applied to one or more features. The request is generated on the client and is posted to a web feature server using HTTP. The web feature server then reads and (in a sense) executes the request. To support transaction and query processing, the following operations are defined in WFS:

- *GetCapabilities*. As in WMS a WFS must be able to describe its capabilities. Specifically, it must indicate which feature types it can service and what operations are supported on each feature type.
- *DescribeFeatureType*. A WFS must be able, upon request, to describe the structure of any feature type it can service.
- *GetFeature*. A WFS must be able to service a request to retrieve feature instances. In addition, the client should be able to specify which feature properties to fetch and should be able to constrain the query spatially and non-spatially.
- *Transaction*. A web feature service may be able to service transaction requests. A transaction request is composed of operations that modify features; that is create, update, and delete operations on geographic features.
- *LockFeature*. A WFS may be able to process a lock request on one or more instances of a feature type for the duration of a transaction. This ensures that serializable transactions are supported.

Based on the operation descriptions above, two classes of web feature services can be defined:

- *Basic WFS*. A basic WFS would implement the GetCapabilities, DescribeFeatureType and GetFeature operations. This would be considered a READ-ONLY web feature service.
- *Transaction WFS*. A transaction web feature service would support all the operations of a basic web feature service and in addition it would implement the Transaction operation. Optionally, a transaction WFS could implement the LockFeature operation.

3. SERVICE ORIENTED ARCHITECTURES (SOA)

Web Services are described by a set of protocols to enable communication between independent software modules that offer their

functionalities in the form of services. Current Web-Services are based on Services Oriented Architectures (SOA) [8]. In a SOA, services are self-contained, modular applications - deployed over standard middleware platforms, e.g., J2EE - that can be described, published, located, and invoked over a network. To support the realization of the service-oriented software paradigm, Web service need to be based on standardized definitions of an interoperability communication protocol, mechanisms for service description, discovery, and composition as well as a basic set of quality of service (QoS) protocols.

The most generic SOA consists of three basis actors (see Figure 1): service requester, service provider and service broker (or service registry provider). Service provider describes services and publishes them in the registry provided by service broker. Once they are published they can be found and bind by service requester, utilizing XML-based protocols (see below). Later and if needed, service requester composes discovered services to obtain the desired functionality and bind/execute them on demand.

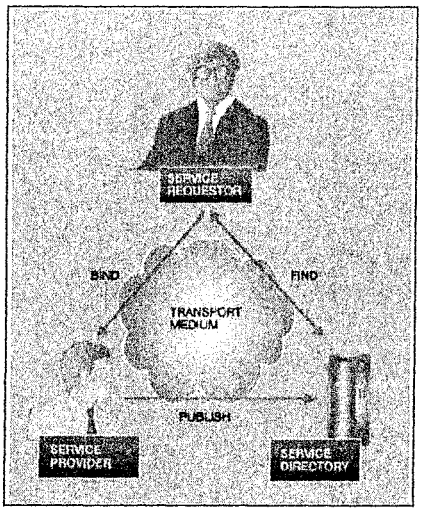

Figure 1. Basic SOA Architecture

The underlying middleware technology has already evolved to conform to the described publish-find-bind scheme: the initial trio of Web service specifications, SOAP[9], WSDL[10], and UDDI[11], provided open XML-based mechanisms for application interoperability (SOAP), service description (WSDL), and service discovery (UDDI). SOAP is now a W3C standard, and WSDL and UDDI are being considered by standard bodies. In order to implement this basic framework in real applications, mechanisms

for service composition and quality of service protocols are required. Several specifications have been proposed in these areas, most notably the Business Process Execution Language for Web Service (BPEL4WS)[12] for service composition, Web service coordination (WS-Coordination) and Web service transactions (WS-Transaction) to support robust service interactions, Web service security (WS-Security), and Web service reliable messaging (WS-ReliableMessaging)[13]. The descriptive capabilities of WSDL can be enhanced by the Web Service Policy Framework (WS-Policy), which extends WSDL to allow the encoding and attachment of QoS information to services in the form of reusable service "policies." All these aspects are critical elements for meaningful services interactions.

The described Web Service protocol stack is shown in Figure 2, from [14]. On the lower level of the stack one finds transport and encoding layers, in the middle level protocols for service description, security, transaction and coordination are located, and, finally, on the top level the protocol stack has the business process composition layer. In [8] more details of the service enabling protocol stack are presented. Moreover, more comprehensive architectures have appeared recently, comprising the basic SOA and usually layered extension covering some of the vital characteristics listed above[15].

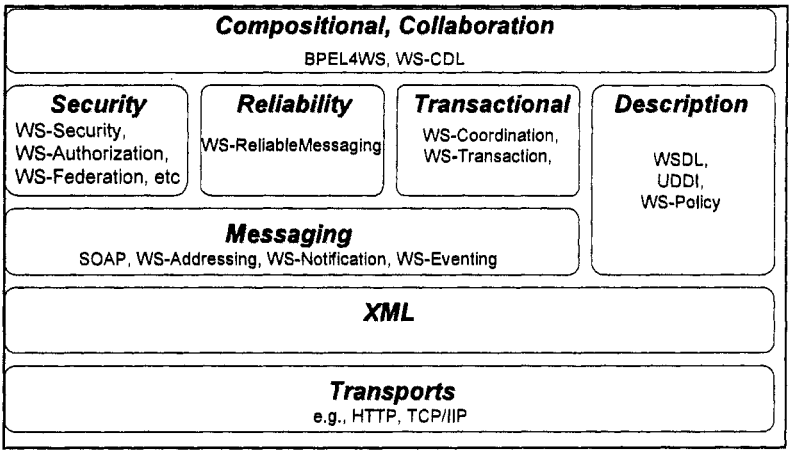

Figure 2. The Web Service protocol stack

4. PROPOSED FRAMEWORK AND CASE STUDY

We propose to take full advantage of the SOA approach in the context of GIS by implementing the operations offered by WMS and WFS following OpenGIS Web Services initiative [16]. To this end we have used Bea Web Logic Server [17] for creating and publishing our specific Web Service interfaces. In particular Bea Web Logic Server provides support for the SOAP communication between server and the client.

We have experimented the proposed architecture in the context of integration of GIS legacy services in a back-office scenario: a user that need to navigate in a spatial database (*location search and feature layer selection*), insert a map (*download of dynamically user-specified raster image centered on searched location*), navigate the image (*pan&zoom*), insert related information in a text document (*legend insertion)* and download locally the selected feature layers in Geographic Markup Language (GML) format (*metadata extraction)*. Traditionally the user would ask the assistance of a GIS technician to produce the overall data. Most of the time she will not be satisfied by the results and interactions with the GIS technician will be iterated.

In our proposed architecture the user can automatically and independently create and insert the current version of the searched geographical data in his/her document using a web service architecture based on OGC specifications. To this end we have design and implemented three main services that provide the user with the appropriate functionalities, namely:

- **TOPService**: service provider of location search by label; this service guides the user in the search of a location from all recognize labels present in the spatial database. The search is implemented in a two-step procedure: first the service searches for a particular string (user input) in the database and delivers the list of all labels that contain the string; second it locates the geographical x,y coordinates associated to a specific label (user input).
- **WMSmapService**: service provider of raster data; it wraps the functionalities defined in the WMS specification in a Web Service interface. Moreover it returns the lists of available layers. The specific supported WMS operations are: "GetCapabilities", "GetMap" and "GetLayers". In particular the last operation is implemented by analysing the XML output of "GetCapabilities" operation.
- **WFSmapService**: service provider of vector data, also in this case we have developed appropriate wrapper WS interfaces to WFS functionalities, namely "GetCapabilities", "DescribeFeature", "GetFeature".

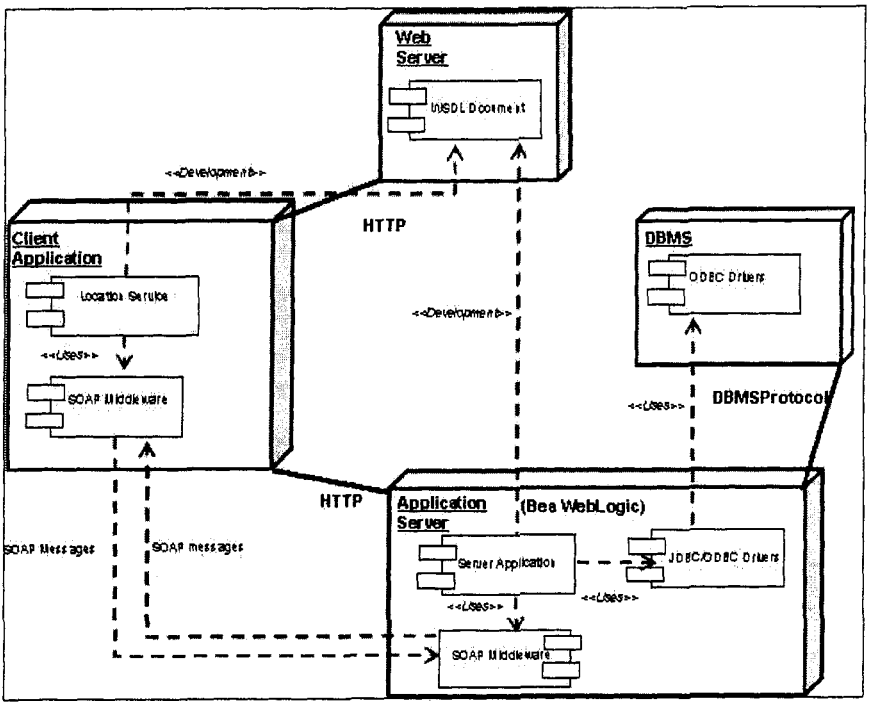

Figure 3. TOPService Component/ Deployment Diagram

In Figure 3 we provide the component and the deployment diagram for the TOPService. In the diagram we can see the various components of our SOA, namely:

- the WSDL file to define the interface of the service
- the use of the SOAP standard for messaging exchanges
- the connection of the SOAP middleware to the appropriate database driver (JDBC/ODBC)
- the DBMS system that implements the SQL query and format the result

Figure 4 provides the component and the deployment diagram for both WMSmapService and WFSmapService. In fact the two services share the same component and message structure and differ only in the specific procedures implementation. Moreover the only difference with the previous diagram for the TOPService is in the connection of the SOAP middleware to the MapServer component driver. This connection is implemented by means of appropriate calls following WMS/WFS specifications. The MapServer then implements the database queries and format the results.

A central role in the proposed architecture is played by the WSDL file created for each implemented service.

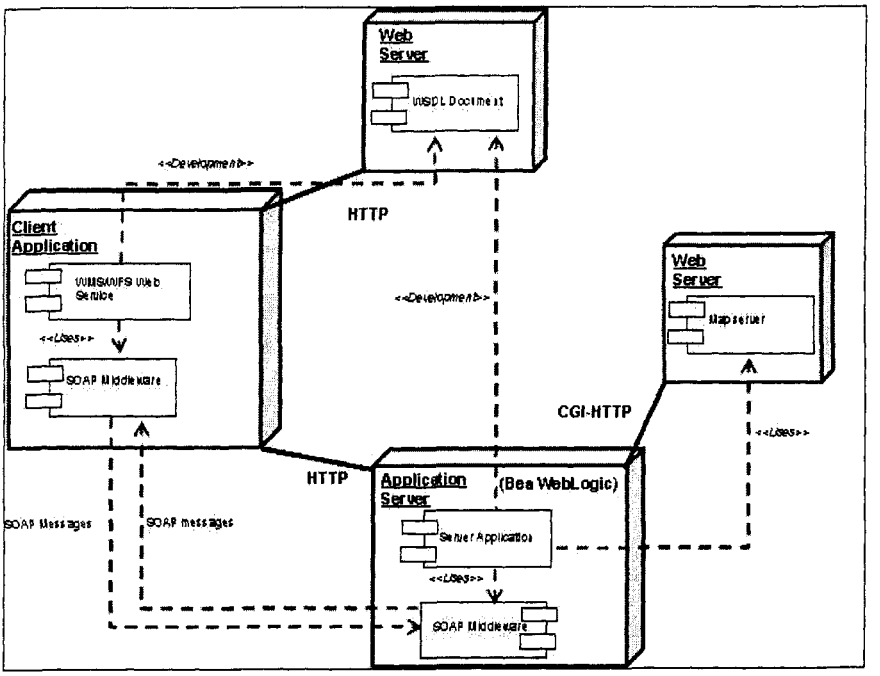

Figure 4. WMSmapService and WFSmapService Component/ Deployment Diagram

The WSDL files are XML documents that describe the mechanics of interacting with the specific web service. Although a web service description in WSDL is written solely from the point of view of the web service (or the service provider that realizes that service), it is inherently intended to constrain both the service provider and the service requester that makes use of that service. This implies that WSDL represents a "contract" between the service requester and the service provider, in much the same way that an interface in an object-oriented programming language, e.g., Java, represents a contract between client code and the actual object itself.

In order to implement a single GIS Web Service, the following steps have to be completed:

- locate an appropriate spatial data base (in our case the GIS of Provincia Autonoma di Trento, PAT)
- define some interesting WMS and WFS services (to this end we have used MapServer for the implementation)
- define the related WSDL files, that expose to the user the offered functionalities
- create server side applications (Web Services providers)
- create client side applications (Web Service requesters)

To support aggregate services, we have first to model the appropriate sequence of processes and then implement them in the service architecture framework. This is possible in two ways: statically, by using proper tools to link the various services needed in the process workflow, or dynamically by describing the workflow in appropriate and adequately expressive languages (such as BPEL4WS [7]).

In our first evaluation of the proposed framework, we have developed a number of aggregate services statically, using the Microsoft Office 2003 Web Services Toolkit to develop the client application. It is worthwhile to note that this choice has been driven by compatibility issues related to the specific back-office development environment used in the local public administration. A similar toolkit is also available in Open Source environment such as Open Office. Figure 5 provides an example of workflow for a typical back-office aggregate service, where the involved services and process flow have been specified. The final result is presented in figure 6, where we show the final outcome of the user request: the inclusion of the requested map in the current document as well as the navigation bar that permits the user a personalization of the graphical appearance.

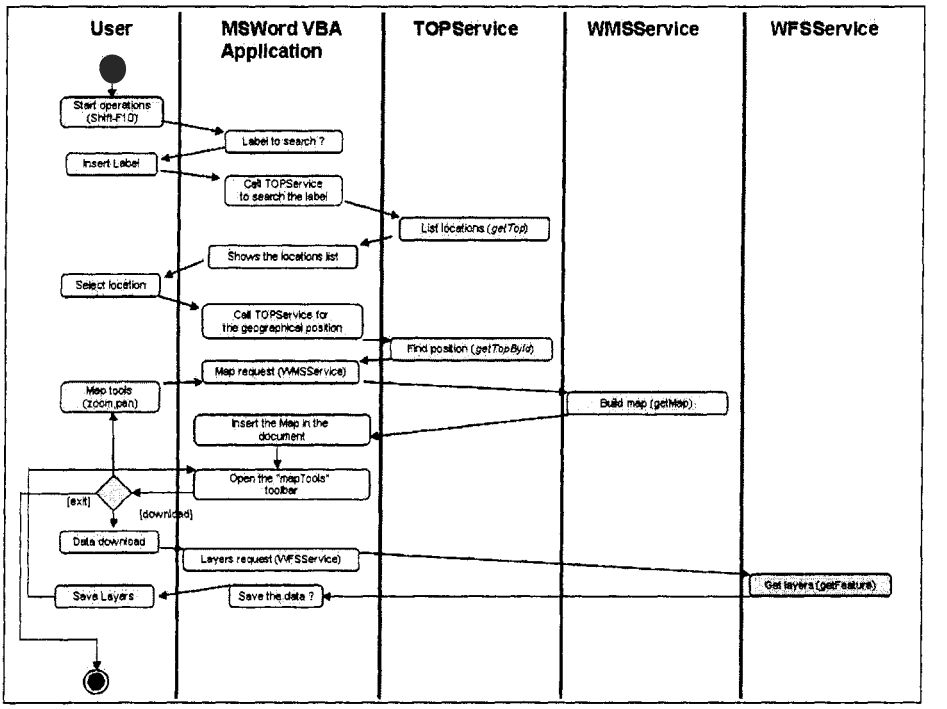

Figure 5. Example of workflow of back-office aggregate service

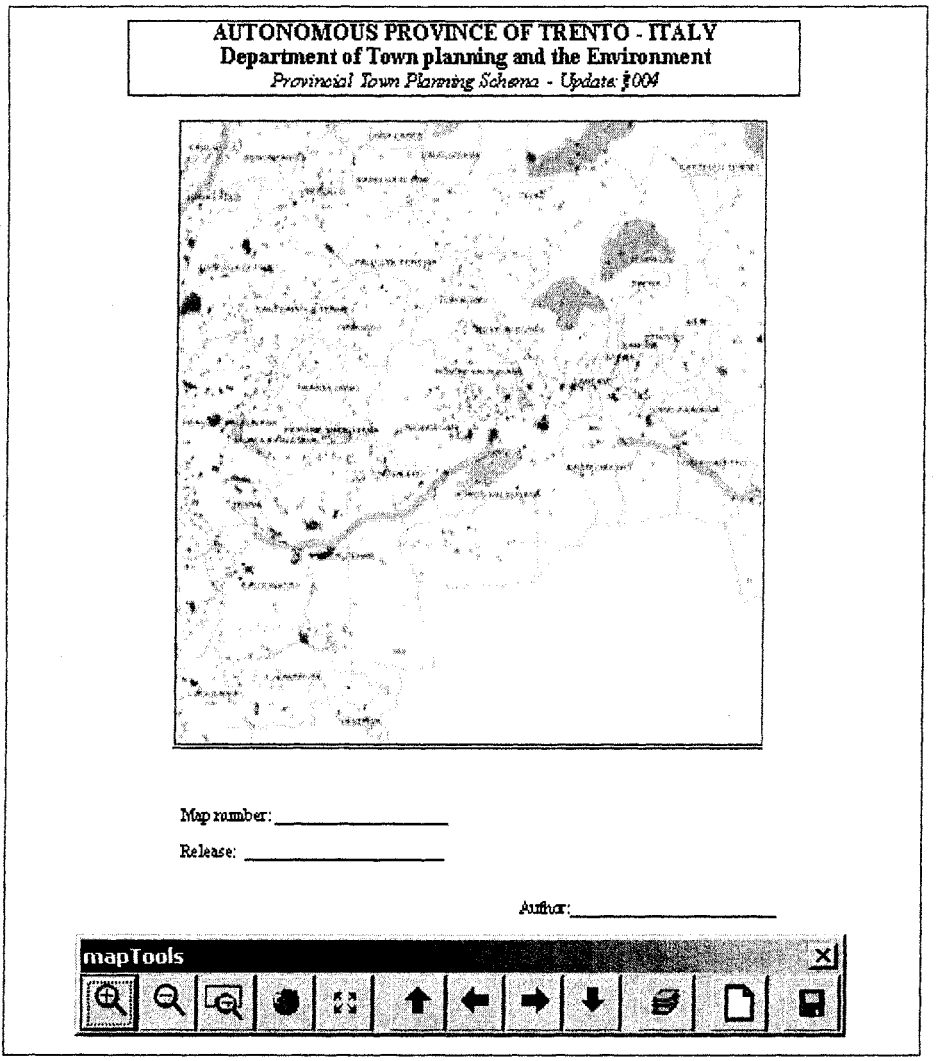

Figure 6. Back-office document with inserted map and navigation tool

5. RELATED WORK

Currently geographical data are shared through stand-alone or web client/server applications based on OGC specifications. A number of geographical servers are present in the market, among which "*MapServer*" (used in our implementation), "*Autodesk MapGuide*", "*ESRI ARCIMS*",

"*CubeWerx CubeSERV WMS*", "*JUMP*", "*OPENMAP*", "*MYSQL Spatial*". Among client implementations we mention: "*Chameleon*", "*Intergraph WMS Viewer*", "*J2ME OGC WMS Client*", "*NASA Web Map Viewer*". For details of the various implementations see [18,20].

On the other hand Web Service implementation of OGC specifications are still under evaluation and standardization from the OGC consortium. One of the few commercial implementation is the ESRI Business Analyst Web Services [20] http://www.esribis.com/solutions/index.html: they comprise a set of centralized services delivered online and delivered to custom Web applications. In particular the user must rely on ESRI geographical database and not on his own data.

OGC is currently investigating SOA in the OpenGis Web Services (OWS) Initiative: an interoperability program designed for rapid development and delivery of proven candidate Web Service specifications into OGC's Specification program which can then be formalized for public release. In particular we recall here a list of relevant initiatives related to our current work:

- OWS 1.2 SOAP Experiment Report [21]: this document describes how OWS services can be ported to Web Services and highlight various issues/problems that have been discovered and need further discussion.
- OWS 1.2 UDDI Experiment [22]: this document lists the design principles, requirements, and experimental results for future versions of a potential OGC – UDDI (Universal Discovery, Description, and Integration) implementation specification.
- OpenGIS® Web Service Architecture [23]: this document is an Interoperability Program Report from the OGC Open Web Services (OWS1.2) Testbed. It specifies and discusses a common architectural framework for OGC Web Services.

6. CONCLUSIONS AND FUTURE WORK

With the evolution of applications dedicated to geographic information and current networking infrastructures, spatial data are required to be accessible to an increasing number of users. Standards bodies are working together to define appropriate interface specifications to support sharing and interoperability between various spatial data platforms. At the same time, the advance of Service Oriented Architectures is contributing to solve a number of limitations of traditional distributed systems. In this work, we have

focused on the use of state-of-the-art Web Services technologies to support and extend current GIS applications.

From a preliminary analysis of the results of our case study we believe that several advantages can be obtained by introducing service oriented architectures into the GIS environment:

- the interoperability between different system can be enhanced
- the availability and usability of geographical information can be improved
- the creation of new valued-added services out of a number of legacy GIS application can be supported

In future work, we will also consider:

- performance issues: since multimedia objects, like maps, can be of relevant size, specific compression algorithms must be considered for data transmission
- security issues: adequate security policies must be established for the deployment of a secure distributed service environment. User can be allowed to query and browse the geographical database, but only authenticated users must be able to actually change the data. This can be achieved with the implementation of the suitable WS-security and WS-Policy framework.

7. REFERENCES

[1] N. Land, Building Europe's Spatial Data Infrastructure (ESDI), Cambridge Conference, 2003. See also http://inspire.jrc.it

[2] for more information on some current Italian projects see. http://labsita.arc.uniroma1.it; http://www.centrointerregionale.it; http://www.intesagis.it

[3] OGC (Open Geospatial Consortium) web site http://www.opengeospatial.org

[4] ISO/TC 211 19128 Geographic information — Web Map Server Interface

[5] P. A. Vretanos, A. Doyle, May 2002, Web Feature Service Implementation Specification, ref. document OGC 02-58, OGC.

[6] S. Cox, A. CuthBert, R. Lake, R. Martell, 2003, Geography Markup Language Implementation Specification OGC 02-023r4, Open GIS Consortium Inc. http://www.opengis.org/techno/documents/02-023r4.pdf

[7] http://www.opengis.org/techno/spec.htm

[8] M.P. Papazoglou, D. Georgakopoulos, Ottobre 2003,Service Oriented Computing, Communication of the ACM, Vol. 46, No. 10.

[9] http://www.w3.org/TR/soap

[10] http://www.w3.org/TR/wsdl

[11] http://www.uddi.org

[12] http://www-106.ibm.com/developerworks/library/ws-bpel

[13] http://www-106.ibm.com/developerworks/library/ws-coord; http://www-106.ibm.com/developerworks/webservices/library/ws-transpec; http://www-106.ibm.com/developerworks/webservices/library/ws-security.

[14] Curbera, F., Khalaf, R., Mukhi, N., Tai, S., and Weerawarana, S., 2003. The next step in web services. In Papazoglou and Georgakopoulos (2003), pages 29–34
[15] M. Papazoglou, J. Dubray, Giugno 2004, A Survey Of Web Service Technology, Technical report of Università di Trento,
[16] http://www.opengeospatial.org/initiatives
[17] http://dev2dev.bea.com
[18] geographical servers implementations:
http://demo.cubewerx.com/demo/cubeserv/cubeserv.cgi;
http://www.vividsolutions.com/jump;
http://www.vividsolutions.com/jump;
http://dev.mysql.com/doc/mysql/en/Spatial_extensions_in_MySQL;
[19] geographical clients implementations: http://www.maptools.org/chameleon;
http://ogc.intergraph.com/webmapviewer/main.asp;
http://www.boege.net/wmsclient_en.html;
http://viewer.digitalearth.gov/
[20] ESRI Business Analyst Web Services: http://www.esribis.com/solutions/index.html
[21] J. Sonnet, C. Savage, 15th January 2003, OWS 1.2 SOAP Experiment Report, Reference document OGC 03-14, Open GIS Consortium Inc. .
[22] J. Lieberman, L. Reich, P. Vretanos, 17th January 2003, OWS1.2 UDDI Experiment, Reference document OGC 02-054r1, Open GIS Consortium Inc. .
[23] J. Lieberman, 18th January 2003, OpenGIS® Web Services Architecture, Reference document OGC 03-025, Open GIS Consortium Inc. .