04JEI00029
1726-2135/1684-8799

www.iseis.org/jei

Journal of Environmental Informatics 3 (2) 77-88 (2004)

The Design and Implementation of an OpenGIS Conforming Feature-Coverage-Map Server Implementation Specification for CORBA

S. F. Zhang* and S. Goddard

Department of Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, Nebraska 68588-0115, USA

ABSTRACT. Two trends in geographic information science today are affecting Environmental Information Systems (EIS): standardization in Geographic Information Systems (GIS), and the application of middleware technology in heterogeneous computing environments. Research problems arise when the two trends meet each other in the EIS domain. In GIS, standardization is important for geospatial data sharing and geospatial service interoperation. The OpenGIS Consortium (OGC) published abstract specifications and partial implementation specifications, which can make diverse geospatial data and geospatial services accessible to conforming applications. For example, the OGC Simple Feature (SF) and Grid/Coverage (GC) geospatial data specifications specify the format and operations on geospatial feature (vector data) and coverage (raster data) as OpenGIS Simple Feature and Grid Coverage; the OGC Web Feature Server (WFS), Web Coverage Server (WCS), and Web Map Server (WMS) can publish and manage geospatial features, coverages, and maps via the Web. However the implementation specifications are still incomplete. With increasing applications of middleware technology, such as Common Object Request Broker Architecture (CORBA), Distributed Component Object Model (DCOM), and Java Remote Method Invocation (RMI), in GIS and EIS, it is necessary to design feature, coverage, and map server implementation specifications for CORBA, DCOM, and Java RMI. This paper presents an integrated feature-coverage-map server implementation specification for CORBA; similar to the OGC WFS, WCS, and WMS implementation specifications, it provides the access to features, coverages, and maps in a CORBA environment. Clients conforming to the implementation specification can interact with a CORBA feature-coverage-map server using the OpenGIS Simple Feature and Grid Coverage. An implementation based on the specification is also presented. The implementation transformed GRASS, an open source GIS, into a CORBA feature-coverage-map server. The evaluation results show that OpenGIS-based interoperability comes at an acceptable cost in terms of performance. The primary contributions of the paper are first that it presents an OpenGIS conforming feature-coverage-map server implementation specification for CORBA, and second that it presents a demonstration implementation and evaluates its performance.

Keywords: CORBA, Coverage Server, EIS, Feature Server, GIS, GRASS, Map Server, OpenGIS

1. Introduction

Computer technologies are becoming more and more important for environmental research. A number of Environmental Information Systems (EIS) have been developed to provide decision support for environmental management (Chang and Wang, 1996; Koschel et al., 1996; Chang et al., 1997a,b; Mailhot et al., 1997; Huang et al., 1999; Soncini et al., 1999; Chang et al., 2001). As a result, the new field of "En- vironmental Informatics" is emerging (Huang and Chang, 2003).

Today two trends in geographic information science are affecting Environmental Informatics. First is the standardization of Geographic Information Systems (GIS). GIS are widely used in EIS (Koschel et al., 1996; Mailhot et al., 1997; Huang et al., 1999), and are effective in handling complicated spatial information, which is essential for many environmental studies, as well as providing platforms for integrating various environmental models, systems and interfaces (Lovejoy, 1997; Huang et al., 1999). Traditional GIS software provides geospatial services with raster and vector geospatial data. However, the geospatial data structure and the geospatial service interfaces are often vendor-dependent or proprietary. Based on the OpenGIS Consortium's (OGC) definitions, there are 5 criteria to decide whether a standard is an open standard or not.

1. The standard is created and owned in an open international, participatory industry process;
2. The standard has free rights of distribution;
3. The standard has an open specification access;
4. The standard should not discriminate against any person or other groups;
5. The standard should be technology neutral.

Under these definitions, a de facto standard established by one company or an exclusive group of companies or by a government is not an open standard, even if it is published and available for use by anyone at no charge (OGC-OPEN, 2003). For example, both ArcInfo and GRASS have their own incompatible format to describe geospatial vector data: ArcInfo uses the shape file while GRASS use a vector map layer (for line and polygon data) and a site list file (for point

* Corresponding author: shzhang@cse.unl.edu

data). Though both formats are de facto standards, they are vendor-dependent and bring difficulty in making different GIS software interoperable. To provide interoperability in accessing multiple, heterogeneous geospatial data and geospatial services, the OGC proposed open standard specifications, including geospatial data presentation and geospatial service interfaces. For example, to standardize the geospatial data format, OGC Simple Feature (SF) (OGC-SF, 1998) and Grid/Coverage (GC) (OGC-GC, 2001) implementation specifications were published for geospatial feature (vector data) and coverage (raster data) respectively. To standardize the geospatial services interaction via the Web, OGC proposed the Web Services-based framework. Under the framework, the Web Feature Server (WFS) (OGC-WFS, 2002) and Web Map Server (WMS) (OGC-WMS, 2001) implementation specifications were published for feature manipulation and map presentation; recently the Web Coverage Server (WCS) (OGC-WCS, 2003) implementation specification was also published for coverage access via the Web. The standard specifications enable users to share large amounts of geospatial data that were originally incompatible and to integrate geospatial services that were originally non-interoperable.

The second trend is the application of middleware technology in heterogeneous computing environments. Generally, environmental questions are inter-disciplinary, which are concerned with a large number of different knowledge domains, such as physical, chemical, biological, and ecological. As a result, autonomous systems from different knowledge domains need to cooperate with each other. However, several facts make the cooperation difficult. First, systems are usually semantically heterogeneous. For example, different environmental data models may not understand each other. Second, systems are usually technically heterogeneous in that they may be implemented under different operating systems and programming languages (Koschel et al., 1996; Goddard et al., 2002). To hide the heterogeneity, open, service-oriented GIS and EIS are replacing the old traditional monolithic GIS and EIS (Jacobsen and Voisard, 1998; Coddington et al., 1998; OGC-ARCH, 2001; Goddard et al., 2003). Under the open, service-oriented architecture, distributed data and functionality can be integrated and cooperate with each other like a service chain. Middleware technologies, such as the Common Object Request Broker Architecture (CORBA), Distributed Component Object Model (DCOM), and Java Remote Method Invocation (RMI), are playing major roles in the construction of such open, service-oriented architectures.

Research problems arise when the two trends meet each other in the EIS domain. For example, given the diversity of distributed computing environments, the OGC's WFS, WCS, and WMS geospatial service implementation specifications are not sufficient for today's GIS and EIS applications. Feature and coverage access, and map presentation specifications should also be available in other distributed computing environments, such as CORBA, DCOM, and Java RMI. The National Agricultural Decision Support System (NADSS) is an example EIS with such requirements. NADSS is being developed with the Risk Management Agency of the United States Department of Agriculture (Goddard et al., 2002; Goddard et al., 2003). As a distributed EIS, the initial focus of the NADSS project is to improve the quality and accessibility of drought related knowledge, information, and spatial analysis for drought risk management. CORBA technology is used as the distributed computing environment in NADSS. To make the geospatial data in NADSS available to external GIS and EIS applications, open and standard-conforming interfaces for the feature server, coverage server, and map server are required.

This paper introduces the design of an integrated feature-coverage-map geospatial service implementation specification for CORBA. The implementation specification contains three server interfaces to features, coverages, and maps, which respectively correspond to the OGC's WFS, WCS, and WMS geospatial service implementation specifications in the Web environment. Moreover, in contrast to the "read-only" WMS specification, it provides a method which can generate new maps based on the client's input data. The rest of this paper first introduces the OGC's Web Services and its WFS, WCS, WMS geospatial service implementation specifications, and CORBA in Section 2, and then describes the design of an OpenGIS conforming client-push feature-coverage-map server specification for CORBA in Section 3. Section 4 introduces an integrated feature-coverage-map server implementation in NADSS based on the specification. To evaluate its performance, a non-OpenGIS conforming CORBA implementation and a CGI-based Web implementation were compared with the OpenGIS conforming CORBA implementation. Section 5 presents the conclusions and our contributions.

2. Background

This section presents related background knowledge. Section 2.1 introduces the geospatial feature, coverage, and map server in GIS. CORBA is briefly introduced in Section 2.2.

2.1. Feature Servers, Coverage Servers, and Map Servers in GIS

There are two kinds of geospatial data in GIS. One is feature (OGC-Abs-Feature, 1999) or vector data, for which the OGC provides the SF geospatial data specification for CORBA, DCOM, and SQL to standardize it. The other is coverage (OGC-Abs-Coverage, 1999) or raster data, for which the OGC provides the GC geospatial data specification for CORBA and DCOM to standardize it. Consequently, a feature server organizes and manipulates features while a coverage server organizes and manipulates coverage in a distributed environment. As a combination of vector data and raster data, a map is a static raster graphic picture of the geospatial data rather than the actual geospatial data itself. A map provides a visual representation for the geospatial data, and a map server is used to organize and publish maps for end users. A map server is necessary in today's GIS and EIS applications because, in general, creating a map is still a time-consuming

process, which depends on both the mapping software and hardware computing capability. For example, the generation of a U.S scale drought map in the NADSS project requires data from thousands of individual weather stations and other geospatial data layers; this task executes for nearly 30 minutes on a dual 1.9 GHz CPU computer. For NADSS users who only want to monitor drought, it is expensive and unnecessary to generate and manage maps themselves.

To make the geospatial data and geospatial services available in a distributed environment, OGC issued the GIS service architecture (OGC-ARCH, 2001) as a general guideline. For the Web, OGC issued an evolutionary, standard-based framework, OGC Web Services (OWS), to allow distributed geo-processing systems to communicate with each other using technologies such as eXtend Markup Language (XML) and Hypertext Transfer Protocol (HTTP). OWS enables interoperation of a variety of online geospatial data sources, sensor-derived information, and geo-processing capabilities. Currently OWS issues the WFS, WCS, and WMS geospatial service implementation specifications. Section 2.1.1 introduces the geospatial service implementation specifications for the OGC WFS, WCS, and WMS. The advantages of similar implementation specifications in other distributed computing environments are presented in Section 2.1.2.

2.1.1. OGC Web Services and Its Implementation Specifications

The OWS uses the HTTP as the communication protocol. HTTP supports two request methods: GET and POST, which use Uniform Resource Locator (URL) to specify the online operations from a Web server. A URL for HTTP GET requests is a URL prefix to which additional parameters must be appended in order to construct a valid OWS operation request. A URL for HTTP POST requests is a complete and valid URL to which clients transmit encoded requests in the body of the POST document. The OGC WMS, WFS, and WCS implementation specifications use either HTTP GET or POST methods to present the features, coverages, or maps to the Web clients. The advantages of OGC Web Services are as follows.

- A well-known port is used. The HTTP port is open for almost every server in the Internet and almost no firewalls will block HTTP port transportation.
- Standard and powerful metadata description. XML is used to describe the meta-information (capability) of the WFS, WCS, and WMS. XML conveys more semantic information than HTML.
- Standard clients already exist. A Web browser is the most common Web client tool for the Web Services.

Figure 1 shows the architecture of WFS, WCS, and WMS. When Web clients issue the feature, coverage, or map request to the Web server, the Web server directs the request to the corresponding feature, coverage, or map server; then the results or exceptions will be returned to the Web clients through the Web server. Since there are no standard OGC SF and GC geospatial data specifications for HTTP-based Web environments (the OGC SF and GC geospatial data specifications are only available for CORBA and DCOM), WFS, WCS, and WMS have to use non-standard formats as the return results. WFS transmits features described by the XML-based Geographic Markup Language (GML). Currently, however, standard Web browsers can only display the text feature information with GML. There is no accepted standard for displaying vector data on client browsers (though recently, scalable vector graphics have been adopted as the standard by the WWW Consortium (Andersson et al., 2003)). The newly issued WCS transmits the coverage to the Web clients. However, since there are no consensus standard coverage encodings via the Web, currently WCS uses either some proprietary formats (such as HDF-EOS) or overly simple formats (GeoTIFF or PNG) to represent the coverage. WMS provides a static map view of the geospatial data to the Web clients in the formats of GIF, PNG, etc.

2.1.2. Feature Server, Coverage Server, and Map Server Implementation Specifications for non-HTTP Environments

The OGC WFS, WCS, and WMS geospatial service implementation specifications use HTTP GET/POST methods between a Web client and Web server. However, open standard specifications for feature servers, coverage servers, and map servers are still undefined for other distributed computing environments. The needs for such specifications include component interaction, server-side efficiency and client-side extensibility. The following paragraphs explain each of these needs.

- Component interaction. A component is a fairly independent computing unit with exposed interfaces. Components can collaborate together to fulfill a special application requirement, such as the service chain introduced in (OGC-ARCH, 2001). The number of component-based GIS and EIS, which are based on an open, service-oriented architecture, is increasing (see for example, Kumar et al., 1997; Coddington et al.,1998; Jacobsen and Voisard, 1998; Tsou and Buttenfield, 1998; Goddard et al., 2002). Component-based middleware uses either CORBA, DCOM, or Java RMI. It is necessary to design feature, coverage, and map geospatial service implementation specifications for those distributed computing environments.
- Server-Side efficiency. As Figure 1 shows, the communication between the client and the application server is not direct. In fact many current interactions between the Web server and the feature, coverage, or map servers use CGI, ASP, JSP or Java Servlets. All of them need the Web server to act as a broker. This is inefficient on the server side. Direct communication between the client and the application server in CORBA, DCOM, or Java RMI environments would be more efficient than WFS, WCS, and WMS. Moreover, thread-based CORBA, DCOM,

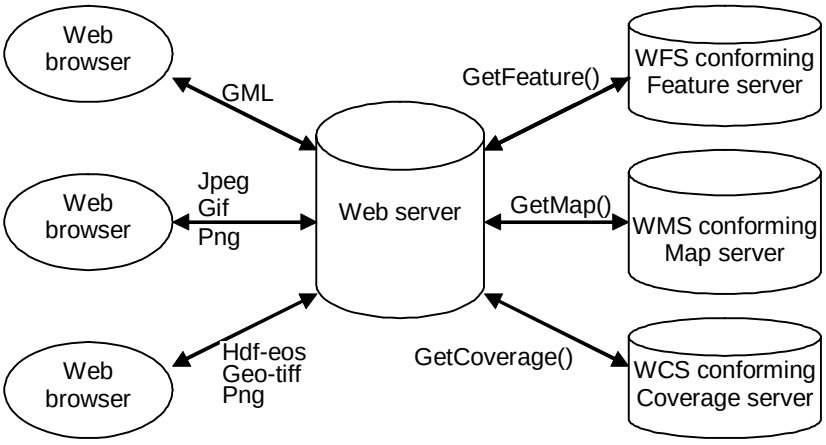

Figure 1. The architecture of Web Feature Server, Web Coverage Server, and Web Map Server.

and Java RMI servers are more efficient than process-based CGI programs, which are commonly used in many of today's Web servers.

- Client-Side Extensibility. In GIS software, the manipulation functions for feature and coverage, such as zoom or pan operations, are common functions. It is not enough to merely display GML-based Simple Feature or pixel-based raster data in a Web browser. A thick client, typically in CORBA, DCOM, or Java RMI environments, can provide more manipulation capabilities. Geotools (Geotools, 2003) and Geo Viewer (Geoviewer, 2003) are examples of such thick clients. Our implementation (introduced in Section 4) also includes a Java client which can display, zoom, and pan the returned results.

With the development of distributed GIS, different middleware technologies are adopted in today's GIS and EIS. As introduced in Section 1, OGC issued the SF and GC geospatial data implementation specifications for CORBA and DCOM, which can be used as the standard geospatial data representations in CORBA, DCOM, or Java RMI environments. As a result, one would expect corresponding OpenGIS conforming feature, coverage, or map geospatial service implementation specifications for the same environments; though no such specification has been issued. With the proposed geospatial service implementation specification introduced in Section 3, these needs can be satisfied in a CORBA distributed computing environment. Moreover, because of the similarity of the CORBA Interface Description Language (IDL), DCOM IDL and JAVA RMI interfaces, the proposed feature-coverage-map server specification (defined with a CORBA IDL file) can be easily ported to the DCOM and Java RMI environments, as demonstrated in Figure 2.

2.2. Introduction to CORBA

CORBA is a well-accepted, mainstream middleware specification. It targets the problems associated with heterogeneity in distributed computing environments. Such heterogeneity is common because platform-dependent computing technology changes over time, e.g., the operating systems and the network technology. CORBA, as a platform-independent computing model and abstraction, can not only hide the heterogeneity between different platforms, but also hide the complexity in the low-level network communication. CORBA provides an Interface Definition Language (IDL) to define the CORBA objects interfaces. The purpose of the IDL is to allow the definition of the object interfaces to be independent of any particular programming languages. In CORBA's architecture, an Object Request Broker (ORB) is responsible for distributing object calls between clients and servers. The object calls can be either static or dynamic. Figure 3 illustrates the role of the ORB.

Many of today's applications, such as GIS and EIS in distributed environments, consist of components from different knowledge-domains which are both technically and semantically heterogeneous. Such heterogeneity can be overcome using CORBA.

3. OpenGIS Conforming Feature-Coverage-Map Server Implementation Specification for CORBA

Many GIS and EIS applications based on CORBA, DCOM, and Java RMI have been developed recently (Koschel et al., 1996; Kumar et al., 1997; Coddington et al., 1998; Jacobsen and Voisard, 1998; Tsou and Buttenfield, 1998; Goddard et al., 2002; Geotools, 2003; Geoviewer, 2003). The trend of GIS and EIS is moving from traditional monolithic systems to open, service-oriented systems (OGC-ARCH,

CORBA IDL	*JAVA Interface*	*DCOM IDL*

module MapSpec {	package MapSpec;	[uuid((7371a240-2e51-11d0-b4c1-444553540000), version(1.0)]
		library MapSpec {
		importlib("stdole32.tlb");

interface MapServer {	public interface MapServer extends
	java.rmi.Remote {
		[uuid(BC4C0AB3-5A45-11d2-99C5-00A02414C655),]
		coclass MapServer {
		interface IMapServer;
		};

string GetMap(in string name);	String GetMap(String name)throws java.rmi.RemoteException;	[uuid(BC4C0AB0-5A45-11d2-99C5-00A02414C655),dual] interface IMapServer :Idispatch { HRESULT GetMap([in]BSTR p1,[out,retval] BSTR *rtn);
};		};
};	*};*	*};*

Figure 2. Similarity of CORBA IDL, JAVA interface, and DCOM IDL for the same interface *MapServer*.

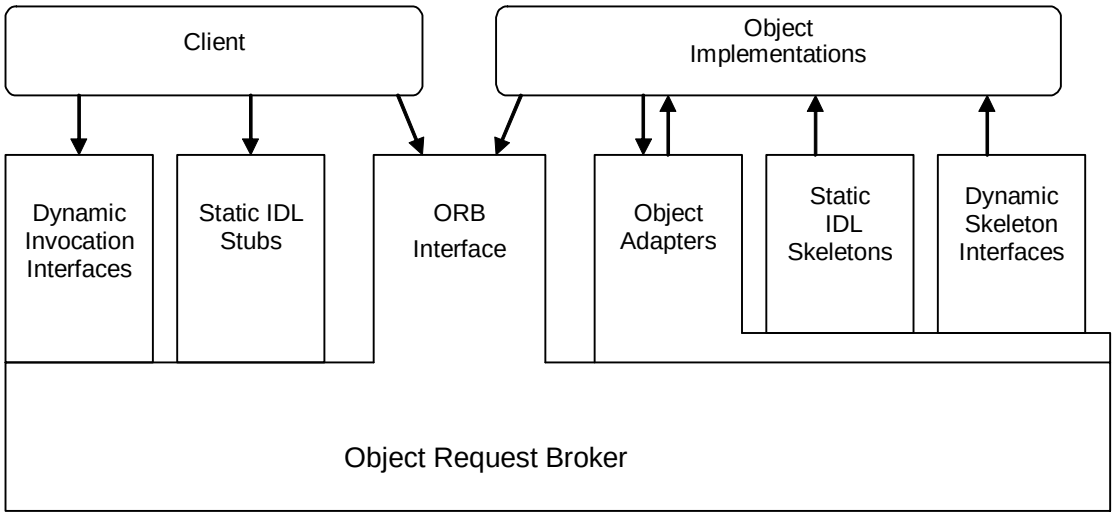

Figure 3. The structure of object request interfaces.

2001). However, a lack of standard feature, coverage, and map server implementation specifications in CORBA, DCOM, and Java RMI environments makes geospatial data sharing and geospatial service interoperation difficult in those distributed environments. OGC WFS, WCS, and WMS implementation specifications provide feature, coverage, and map access methods through HTTP. However, there should exist similar servers that support other non-HTTP distributed environments. Using CORBA, DCOM, or Java RMI technologies, the design and implementation of feature, coverage, and map servers can be more efficient and flexible than HTTP-based servers. First, CORBA, DCOM, and Java RMI provide more mature and efficient ways to implement distributed application systems than HTTP GET/POST methods; second OGC already provided the SF and GC geospatial data implement specifications for CORBA and DCOM, which can be elegantly integrated into the feature, coverage, and map geospatial service implementation specifications for CORBA, DCOM, and Java RMI.

Section 3.1 introduces the interface design of the proposed feature-coverage-map server implementation specification. It refers to WFS, WCS, and WMS implementation specifications and extends support to the CORBA environment. Section 3.2 introduces the client-push map server concept in the proposed specification. Section 3.3 shows the relationship between the proposed feature-coverage-map geospatial service implementation specification, and the OGC SF and GC geospatial data implementation specifications. The full interface definition is given in Appendix A.

3.1. The Definition of the CORBA Feature, Coverage, and Map Server Interfaces

OGC WFS, WCS, and WMS implementation specifications are the existing standards for the feature, coverage, and map access via the Web. Thus it makes sense to use them as the basis for the design of specifications in other distributed computing platforms, as we have done for the CORBA environment. In addition to this, we have added some new features which are described in Sections 3.2 and 3.3.

To simplify the proposed implementation specification, we combined the feature, coverage, and map server implementation specifications into one implementation specification. As a result, the corresponding IDL has three interfaces under one module, each for the feature, coverage, and map server. A conforming client can retrieve the OpenGIS Simple Feature, Grid Coverage, and maps from the feature-coverage-map server implementation. To create a CORBA feature-coverage-map server implementation specification, it is necessary to define a CORBA IDL interface similar to the XML DTD (or schema) of WFS, WCS, and WMS implementation specifications. Currently for each interface, there are two operations in the proposed specification (see Appendix A). First is the *GetCapability()* operation; it returns two sections of meta information about the server to the client: the *service* and the *capabilities*. The *service* section provides a description of the server itself, such as the server name, its provider contact information, fees imposed by the provider, etc. The *capabilities* section provides meta information about what Simple Feature, Grid Coverage and maps can be provided by the server. The second operation is the set of geospatial data access functions; for example, the operation *GetFeature()* from WFS, the operation *GetCoverage()* from WCS, and the operation *GetMap()* from WMS. Operation *GetFeature()* returns the requested OpenGIS Simple Feature, *GetCoverage()* returns the requested OpenGIS Grid Coverage, while operation *GetMap()* returns the requested maps, also as OpenGIS Grid Coverage.

3.2. Client-Push Map Server

The WMS implementation specification defines a client-pull geospatial data server, in which clients can only get geospatial data from the servers based on their capabilities. Hence, a client-pull map server is, in a sense, a "read only" map server in that clients cannot provide their own data to generate the desired map. Though the OGC's Styled Layer Descriptor implementation specification lets users define the presentation style of the map layer (OGC-SLD, 2001), which allows the users to have "write" capability to create new presentation styles for a map layer, users still cannot generate new layers with their own data.

Client-push map servers, on the other hand, are more active than client-pull map servers. With a client-push map server, users can make use of the GIS software in the map servers to generate their desired maps. Considering that current commercial GIS software packages are expensive, and they usually require high-performance computing for interactive map generation, a client-push map server is beneficial for map users. For example, in NADSS, the map server is used more to generate maps than to query maps. Users provide site-based data to the map server and get an interpolation map based on their input data. To implement such a client-push map server, a new operation *GenerateMap()* is added to provide the map generation service in the map server interface. The new operation uses the interpolation functions of the underlying GIS map server to generate the map. Also, two new service elements are added to the proposed implementation specification to support map generation operations. First, a new layer attribute boolean variable "*usercreate*" is added in the service element structure "*layer_attributes*", which indicates whether such a layer can be generated based on the user's input data or not. Second, a new data structure "*SiteValuePair*" is also defined for the client to hold and transmit the input point data (see the Appendix A). An example using *GenerateMap()* is given in Section 4.2.

3.3. OpenGIS Geospatial Data Presentation in the CORBA Feature-Coverage-Map Server

The OGC Web Services framework standardized the access interfaces of WFS, WCS, and WMS; however, limited by HTTP, the returned results of WFS, WCS, and WMS are not OpenGIS conforming formats. For example, the operation *GetMap()* in the WMS implementation specification is an HTTP address to a GIF or PGN file. Similarly, the returned

result of operation *GetFeature()* in the WFS implementation specification is the feature presentation in GML format; and the returned result of operation *GetCoverage()* in the WCS implementation specification are either proprietary formats or overly simple formats. The OGC already standardized geospatial data formats with the SF and the GC geospatial data implementation specification under CORBA and DCOM environments. The SF implementation specification defines interfaces for the feature (vector data) and the GC implementation specification defines interfaces for the coverage (raster data). To conform to the OGC's standard geospatial data spe- cifications, our proposed specification follows the OGC SF specification as the returned vector data format and the OGC GC specification as the returned raster data format. We also consider a raster-based map to be an OpenGIS Grid Coverage (OGC-Abs-Coverage, 1999). Thus, the OGC GC specification is also adopted as the returned raster-based map format. By that we assure the transmitted geospatial data are OpenGIS conforming geospatial data. As depicted in Figure 4, the proposed CORBA feature-coverage-map server implementation specification uses the OGC SF and GC geospatial data implementation specifications as the standard format for the geospatial data.

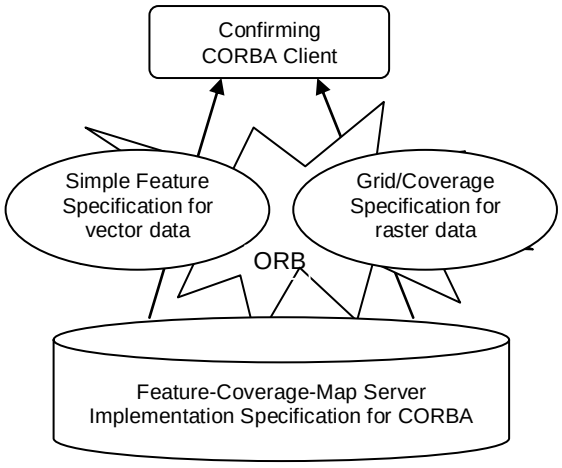

Figure 4. Feature-Coverage-Map server implementation specification and its relation with OGC SF and GC geospatial data implementation specifications.

4. CORBA Feature-Coverage-Map Server Implementation in NADSS

NADSS is a component-based distributed EIS (Goddard et al., 2002; Goddard et al., 2003; Zhang and Goddard, 2003). It provides geospatial related data in the form of climate data (e.g., temperature and precipitation) for agricultural models, information in the form of drought indices, and knowledge in the form of exposure analysis (e.g., the impact of a natural hazard). Two methods are used to make the data, information and knowledge available to end users. First, Web-based interfaces are provided by which users can access and download data, information and knowledge from the Web site. Second, a CORBA-based Java applet client is provided. Users automatically download the applet from the NADSS Web site to access the data, information and knowledge. By providing an OpenGIS conforming feature-coverage-map server, conforming clients can have direct access to the OpenGIS Simple Feature and Grid Coverage geospatial data in NADSS. Section 4.1 introduces the architecture and the implementation of a CORBA feature-coverage-map server with GRASS. Section 4.2 introduces an application example of a CORBA feature-coverage-map server in NADSS. The evaluation of the implementation is presented in Section 4.3.

4.1. Feature-Coverage-Map Server Architecture and Implementation with GRASS

Figure 5 shows the architecture we have used to transform a non-interoperable GIS into an OpenGIS conforming system. By adding a feature-coverage-map server that conforms to the proposed implementation specification, an external conforming client can get OpenGIS Simple Feature and Grid Coverage geospatial data from the non-OpenGIS conforming data sources. To implement the architecture, two necessary elements are needed for the feature-coverage-map server.

First, we need the implementation of the OGC SF and GC geospatial data specifications for CORBA. As indicated in Section 3.3, the feature-coverage-map server uses the OGC SF and GC geospatial data specifications to represent the geospatial data. At present we have implemented the OGC SF specification for CORBA with the following geometries: *Point, LineString, LinearRing* and *LinearPolygon*. We also partially implemented the OGC GC interfaces for CORBA. The implementations are used in the feature-coverage-map server to wrap the underlying GIS geospatial vector and raster data and return them as OpenGIS Simple Features and Grid Coverages.

The second important part of the server is the underlying GIS software, which provides the real geospatial data and geospatial processing functions. GRASS (Neteler and Mitasova, 2002), an open-source GIS, is the underlying GIS software used in NADSS. GRASS provides powerful raster and vector data processing functions. However, it is still a traditional command-oriented GIS. Unlike ArcInfo, which already provides an internet-based solution to Web users (ESRI, 2002), GRASS currently lacks the capability to expose directly its geospatial processing functions and geospatial data in a distributed environment. GRASSLinks is a popular method to expose GRASS functions and GRASS geospatial data through the Web (Huse, 1995). Since GRASSLinks is a CGI-based method, it is neither efficient nor flexible because of the drawbacks of CGI. However, its popularity demonstrates the need to adapt GRASS to a distributed environment.

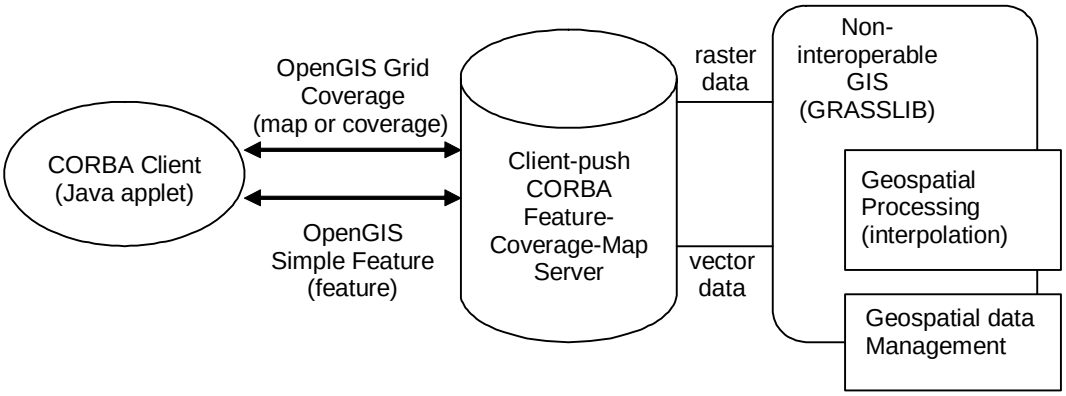

Figure 5. OpenGIS conforming feature-coverage-map server implementation in NADSS.

It is necessary to change GRASS from its current command-oriented style to a component style, which can provide the GRASS functions through an object-oriented method in a distributed environment. In NADSS, we separate the GRASS functions into two categories: geospatial data processing and geospatial data management. Within the geospatial data processing part, GRASS map generating functions are provided as a client-push map server. While within the geospatial data management part, GRASS vector data are served as OpenGIS Simple Features and raster data are served as OpenGIS Grid Coverages. We developed GRASSLIB (Wu et al., 2004), a package to wrap the GRASS commands as shared library functions, which can be used in CORBA, DCOM, and Java environments. Currently there are sixteen GRASS commands that can be called directly from the GRASSLIB library, including the necessary mapping functions used in the CORBA server.

Koschel did similar work by wrapping GRASS with CORBA objects (Koschel and Wiesel, 1997). The difference is that he used GRASS only as a readable GIS server while in NADSS we make use not only of the geospatial data management functions but also of the geospatial data processing (interpolation) functions. More importantly, in (Koschel and Wiesel, 1997) the interfaces don't conform to OGC specifications, so it is impossible for OpenGIS conforming clients to access the geospatial data. While in NADSS, a conforming client can access the OpenGIS Simple Feature and the Grid Coverage from the feature-coverage-map server.

4.2. CORBA Feature-Coverage-Map Server Application in NADSS

One of NADSS's tasks is to use the climatic data to generate drought indices, such as the Standard Precipitation Index (SPI) (McKee et al., 1993) and the Palmer Drought Severity Index (PDSI) (Palmer, 1965). The SPI is a precipitation based index that can be used to monitor drought conditions on a variety of time scales; this temporal flexibility allows the SPI to be useful in monitoring both short-term agricultural and long-term hydrological droughts. The goal of SPI is to assign a single value to the precipitation levels and then compare across regions of different climates. The SPI represents the number of standard deviations that the observed precipitation value deviates from the mean of a normalized gamma distribution for the site. The PDSI is an index that represents the moisture departure for a region, implementing a simple supply-and-demand model for a water balance equation. Thus unlike the SPI, the PDSI is based on more than just precipitation. The value of the PDSI is reflective of the how the soil moisture compares with normal conditions. A given PDSI value is a combination of the current conditions and the previous PDSI value, so the PDSI also reflects the progression of trends, whether it is a drought or a wet spell.

Both the SPI and the PDSI are site-based point information. To compute and display the SPI or PDSI regionally, we need to use spatial interpolation methods to generate county, state, or even national level SPI or PDSI geospatial data. Before using NADSS, climatologists had to manually compute and gather SPI or PDSI values for individual weather stations, then use GIS software (GRASS or ArcInfo) to interpolate the results. The whole process was error-prone and time-consuming (normally 1-2 days for one state). By using the feature-coverage-map server, NADSS can automate the whole process and generate the corresponding OpenGIS Simple Feature and Grid Coverage in seconds. Figure 6 demonstrates how to generate a SPI map for Nebraska, U.S.A. First the Java Client calls the SPI server to get the necessary SPI site information; second the client calls *GenerateMap()* of the feature-coverage-map server to interpolate the site data and retrieve the SPI map as an OpenGIS Grid Coverage. The generated SPI maps provide a visual representation of the drought information for a region, which can be used as an agriculture decision support tool. The code fragment below shows the Java client calling the feature-coverage-map server

to retrieve a map:

```
"
//get the object reference of mfcserver and assign values to the get-
maprequest and nebraska_sites
MFCServer    mfcserver;
MFCSpec.GetMapRequest getmaprequest = new
MFCSpec.GetMapRequest();
MFCSpec.SiteValuePair[] nebraska_sites    ;
gc.GC_Coverage return_coverage;
……
//call the GenerateMap
try {
        return_coverage = mfcserver.GenerateMap (getmaprequest,
nebraska_sites)
} catch (LayerNotValid exceptionfromserver){
……
}
"
```

4.3. Performance Evaluation of the CORBA Feature-Coverage-Map Server

The OGC's implementation specifications make different GIS software interactions possible. However, interoperability comes at a price in terms of performance. Currently CORBA's specification does not support object serialization. It cannot transmit an object by its value instead of its reference. Since OGC conforming clients can only get references to the OGC Simple Features and Grid Coverage objects, they need further communication with the feature-coverage-map server to get the geospatial data. This communication is sometimes heavy, which may cause poor performance especially when bandwidth is limited.

To evaluate the performance of our implementation, we compared map generation functions from an OpenGIS conforming CORBA server, a non-OpenGIS conforming CORBA server, and the CGI-based NADSS Web Map Server (which used GRASSLinks to provide web-based map generation functions). All three servers used the same weather station input data for Nebraska, then generated a state-level SPI map (as displayed in Figure 6) which combined both vector data (county boundaries and state boundary lines) and raster data (an interpolation result of the point data), and returned the raster data to the clients. The OpenGIS conforming COR- BA client first retrieved the OpenGIS Grid Coverage CORBA object and then retrieved data from the Grid Coverage CORBA object; the non-OpenGIS conforming CORBA client retrieved matrix-based raster data; and the Web Client retrieved a PNG file from the NADSS Web server. The unit of process time is in seconds and we tested the average process time for three different spatial resolutions (3000 m [low resolution], 1000 m, 400 m [high resolution]). The different resolutions affect the interpolation result size and consequently the transferred raster data size (the greater the resolution, the larger the raster data size).

Table 1 presents the average call times and standard error from the test samples of each server at each resolution. The average call times are plotted in Figure 7 to provide a visual comparison of their performance. From the results, we found that the CGI-based Web client request took more time than others, and the non-OpenGIS conforming CORBA client request is the most efficient. However, the performance impact of conforming to the specification is acceptable, and it will be reduced when the Object Management Group publishes a CORBA specification that supports objects transmitted by value.

5. Conclusions

As indicated by Huang and Chang in (Huang and Chang, 2003), the GIS standardization proposed by OpenGIS, GRID computing in heterogeneous environments, and large-scale databases combined with artificial intelligence techniques will greatly impact the new field of "Environmental Informatics." In this paper, we show it is useful and practical to apply standardization in GIS and middleware technology to GIS and EIS. At present, standardization in GIS is still under development. The feature, coverage, and map server implementation specifications are undefined for distributed computing environments such as CORBA, DCOM, and Java RMI. OGC WFS, WCS, and WMS geospatial service implementation specifications are only available for HTTP-based Web Services environments. WMS "specifies implementation and use of those WMS operations in the HTTP Distributed Computing Platform (DCP). Future versions may apply to other DCPs" (OGC-WMS, 2001). The increasing number of GIS and EIS applications under open, service-oriented architectures, need similar implementation specifications, such as for CORBA, DCOM, and Java RMI, to interoperate with each other in other distributed environments.

To satisfy the needs for feature, coverage, and map servers in CORBA environments, the work introduced in this paper makes the following contributions. First, it presents and reports on the implementation of a feature-coverage-map server implementation specification for CORBA, which can also be ported to DCOM and Java RMI environments. The specification provides feature, coverage, and map services for CORBA environment, which is similar to WFS, WCS and WMS in HTTP-based Web. Moreover, it also allows users to generate maps with their own data. Unlike WMS, WFS, and WCS, which could not adopt the OGC SF and GC geospatial data implementation specifications, the geospatial data in the proposed feature-coverage-map server implementation specification conform to the OGC SF and GC implementation specifications.

Second, to implement the proposed specification, we used GRASS as the underlying GIS to provide the geospatial data management and geospatial data processing. Traditional GIS software, like GRASS, is command-oriented, and unsuitable for use in a component-based distributed computing environment. This limitation has prevented their wide spread usage in today's distributed GIS applications. Thus, we trans-

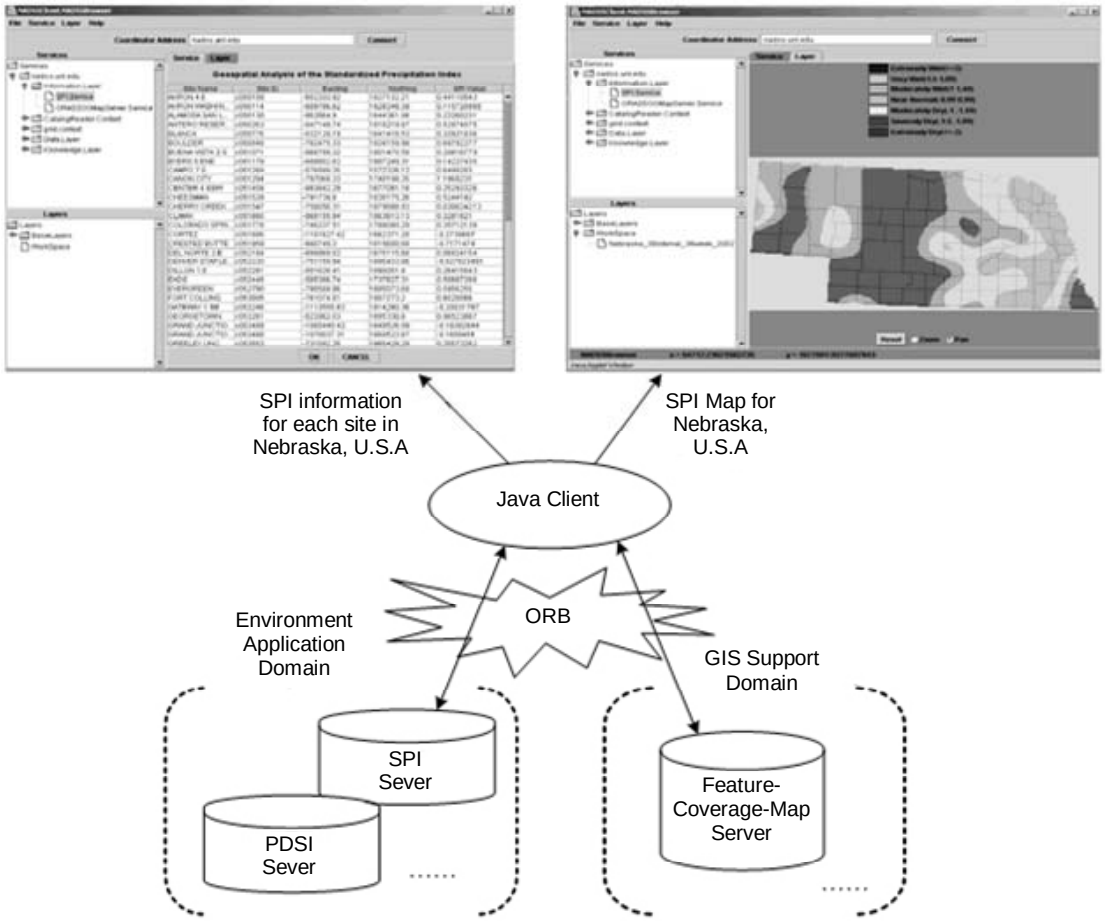

Figure 6. The example of Standard Precipitation Index map generation and its screen snapshots.

Table 1. The Mean Call Time in Seconds and the Standard Error of the Test Samples for the Non-OpenGIS Conforming CORBA Server, the OpenGIS Conforming CORBA Server, and the NADSS CGI-based Web Server

	Non-OpenGIS CORBA server (mean/standard error)	OpenGIS CORBA server (mean/standard error)	CGI-based Web server (mean/standard error)
Low resolution (3000m)	0.5833956s/0.034122	1.1099574s/0.059338	4.9544212s/0.271229
Medium resolution (1000m)	1.5377292s/0.00877	3.808422s/0.139247	7.2575044s/0.24269
High resolution (400m)	13.6558648s/1.17814	18.8938056s/1.158844	37.4011828s/1.21313

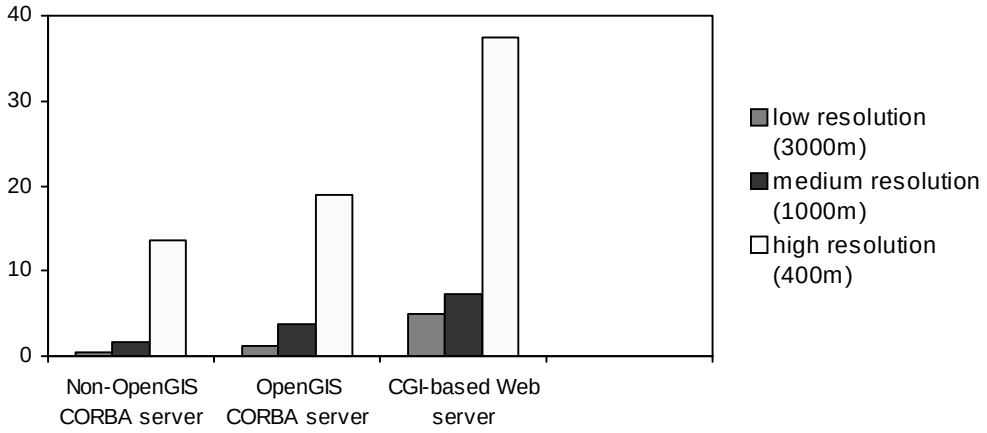

Figure 7. The evaluation results of the CORBA feature-coverage-map server (the y-axis time units are seconds).

formed GRASS into a shared library (GRASSLIB) within a CORBA component to create a feature-coverage-map server based on the proposed specification. To the best of our knowledge, transforming GRASS into OpenGIS conforming CORBA components has never been done before. The proposed implementation specification and its implementation are demonstrated using the NADSS project and SPI and PDSI drought indices. For the specification and its implementation, one of the biggest advantages is providing interoperation to a GIS within CORBA environments; another advantage of the implementation is that it transforms a traditional GIS into a component for distributed environments. A disadvantage is that the standard brings complexity to the implementation; for example the OGC SF specification has nearly 70 interfaces, which is a big burden for developers. Similarly, performance is decreased as compared to a non-standard CORBA implementation. Our experience working with the standard is that the advantages far outweigh the disadvantages.

Acknowledgments. This work was supported, in part, by a grant from NSF (EIA-0091530) and a cooperative agreement with USADA FCIC/RMA (2IE08310228).

References

Andersson, O., et al. (2003). Scalable Vector Graphics (SVG) Version 1.1. http://www.w3.org/TR/SVG11 (accessed by Dec. 26, 2003).

Chang, N.B. and Wang, S.F. (1996). The development of an environment decision support system for municipal solid waste management. *Comput., Environ. Urban Syst.*, 20(3), 201-212.

Chang, N.B., Lu, H.Y. and Wei, Y.L. (1997a). GIS technology for vehicle routing and scheduling in solid waste collection systems. *J. Environ. Eng.*, 123(9), 901-910.

Chang, N.B., Wei, Y.L., Teng, C.C. and Kao, C.Y. (1997b). The design of a GIS-based decision support system for chemical emergency preparedness and response in an urban environment. *Comput., Environ. Urban Syst.*, 21(1), 67-94.

Chang, Y.C., Chang, N.B. and Ma, G.D. (2001). Internet web-based information system for handling scrap vehicles disposal in Taiwan. *Environ. Model. Assess.*, 6(4), 237-248.

Coddington, P.D., Hawick, K.A., Kerry, K.E., Mathew, J.A., Silis, D.L., Webb, P.J., Whitbread, C.G., Irving, M.W., Grigg, R. and Jana, K.T. (1998). Implementation of a Geospatial Imagery Digital Library using Java and CORBA, in *Proc. of Technologies of Object-Oriented Languages and Systems (TOOLS) Asia'98*, Beijing, China, pp. 280-290.

ESRI (2002). ArcIMS 4 Architecture and Functionality, An ESRI White Paper.

Geotools. Open Source Mapping Toolkit. http://geotools.sourceforge.net/ (accessed Dec. 26, 2003).

Geo Viewer. http://elib.cs.berkeley.edu/gis/ (accessed Dec. 26, 2003).

Goddard, S., Zhang, S., Waltman, W., Lytle, D. and Anthony, S. (2002). A software architecture for distributed geospatial decision support systems, in *Proc. of 2002 National Conference for Digital Government Research*, Los Angeles, CA, pp. 45-52.

Goddard, S., Harms, S., Reichenbach, S., Tadesse, T. and Waltman, W. (2003). Geospatial decision support for drought risk management. *Commun. ACM*, 46(1), 35-37.

Huang, G.H., Liu, L., Chakma, A., Wang, X.H. and Yin, Y.Y. (1999). A hybrid GIS-supported watershed modeling system. *Hydrol. Sci. J.*, 44, 597-610.

Huang, G.H. and Chang, N.B. (2003). Perspectives of environmental informatics and systems analysis. *J. Environ. Inform.*, 1(1), 1-6.

Huse, S. (1995). GRASSLinks: *A New Model for Spatial Information Access in Environmental Planning*, Ph.D. Dissertation, Department of Landscape Architecture, University of California, Berkeley.

Lovejoy, S.B. (1997). Watershed management for water quality protection: Are GIS and simulation models the answer?. *J. Soil Water Conserv.*, 52, 103-110.

Jacobsen, H.A. and Voisard, A. (1998). CORBA-based interoperable geographic information systems, in *Proc. of Euro Parallel and Distributed Systems Conference*, Vienna.

Koschel, A., Kramer, R. and Nikolai, R. (1996). A federation architecture for an environmental information system incurporating GIS, the World Wide Web, and CORBA, in *Proc. of NCGIA conference on GIS and Environmental Modeling*, 1996.

Koschel, A. and Wiesel, J. (1997). GIS Operations under CORBA. http://www.lfu.baden-wuerttemberg.de/lfu/uis/globus_direkt/glo

bus3/223-fzic/gl3.html (accessed Dec. 26, 2003).

Kumar, K., Sinha, P. and Bhatt, P.C.P. (1997). DoGIS: A distributed and object oriented GIS, in *Proc. of the Seventh International Symposium on Spatial Data Handing*, Netherlands. pp. 263-275.

Mailhot, A., Rousseau, A.N., Massicotte, S. and Dupont, J. (1997). A watershed-based system for the integrated management of surface water quality: the GIBSI system. *Water Sci. Technol.*, 36, 381-387.

McKee, T.B., Doesken, N.J. and Kleist, J. (1993). The relationship of drought frequency and duration to time scales, in *Proc of 8th Conference on Applied Climatology*, pp. 179-184.

Neteler, M. and Mitasova, H. (2002). *Open Source GIS: A GRASS GIS Approach*, Kluwer Academic Publisher, ISBN: 1-4020-7088-8.

OpenGIS Consortium Inc. (2003). The Importance of Going "Open", An OpenGIS Consortium White Paper. http://www.opengis.-org/docs/2003/20030923_OpenWP.pdf (accessed Nov. 26, 2003).

OpenGIS Consortium Inc. (1999). *The OpenGIS Abstract Specification Topic 5: Features*, OGC Project Document No.99-105.

OpenGIS Consortium Inc. (1999). *The OpenGIS Abstract Specification Topic 6: The Coverage Type and its Subtypes*, OGC Project Document No. 99-106.

OpenGIS Consortium Inc. (2001). The OpenGIS Abstract Specification Topic 12: The OpenGIS Service Architecture, OGC Project Document No. 02-112.

OpenGIS Consortium Inc. (2001). *OpenGIS Grid Coverage Implementation Specification*, OGC Project Document No. 01-004.

OpenGIS Consortium Inc. (1998). *OpenGIS Simple Features Implementation Specification for CORBA*, OGC Project Document No. 99-054

OpenGIS Consortium Inc. (2001). *OpenGIS Styled Layer Descriptor Draft Candidate Implementation Specification*, OGC Document No. 01-028.

OpenGIS Consortium Inc. (2003). W*eb Coverage Service Implementation Specification*, OGC Document No. 03-065r6.

OpenGIS Consortium Inc. (2002). *Web Feature Service Implementation Specification*, OGC Document No. 02-058.

OpenGIS Consortium Inc. (2001). *Web Map Service Implementation Specification*, OGC Document No. 01-068r2.

Palmer, W.C. (1965). *Meteorological Drought*, Research Paper No. 45, US Department of Commerce Weather Bureau, Washington DC, USA.

Soncini-Sessa, R., Luca, V. and Enrico, W. (1999). Twole: A software tool for planning and management of water reservoir networks. *Hydrol. Sci. J.*, 44, 619-632.

Tsou, M.H. and Buttenfield, B.P. (1998). Client/Server components and metadata objects for distributed geographic information servers-GIServices, in *Proc. of GIS/LIS'98*, Fort Worth, TX.

Wu, X. and Goddard, S. (2004). Development of a Component-based GIS using GRASS, in *Proc. of FOSS/GRASS 2004*, Bangkok, Thailand.

Zhang, S. and Goddard, S. (2003). 3CoFramework: A component-based framework for distributed applications, in *Proc. of International Conference on Software Engineering Research and Practice*, Las Vegas, pp. 398-406.