Web Service-based Scientific Data Grid
(Sci-Data Grid)

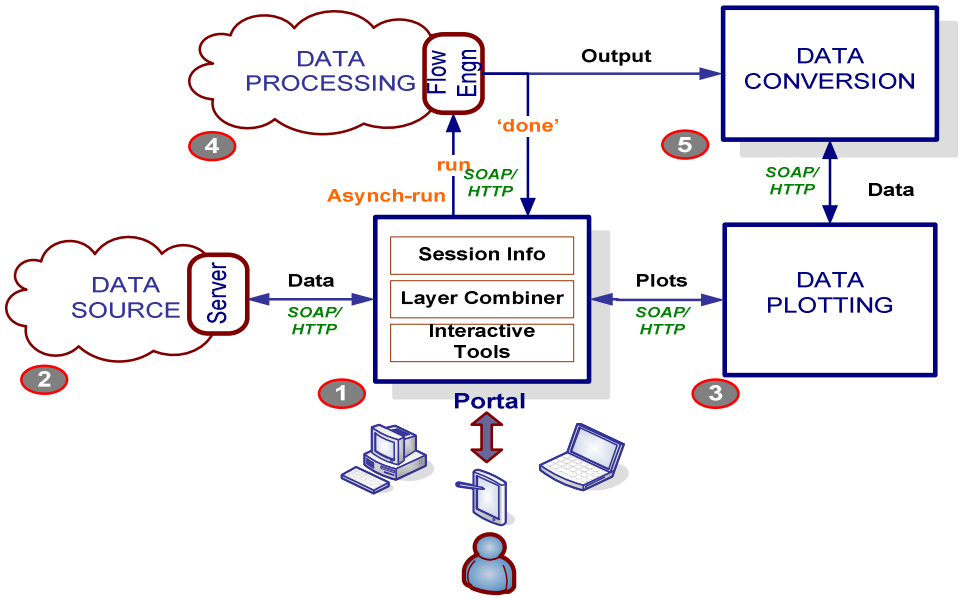
1. Introduction

Sci-Data Web Service Grid framework provides interoperability of services and data across the organizations and providers. Since this framework is based on Web Services, it is easy to integrate new data and processing services into it. All the services and data provider components in the system might be geographically distributed or locally deployed. They are built around the principles of Service Oriented Architecture (SOA). Such systems unify distributed services through a message-oriented architecture, allowing loose coupling, scalability, fault tolerance, and cross-organizational service collections [4]. Web Service standards [5] are a common implementation of SOA ideals, and Grid computing has converging requirements [1, 2]

All the services in the architecture fall into one of the four general categories. These are data services (Data Sources), processing services (Data Processing), conversion and utility services (Data Conversion and Data Plotting), and registry-catalog services. We also can enhance the architecture by adding WS-Context [16] (utility service) to enable asynchronous run and Registry-Catalog service to enable service registry and discovery.

There are two types of workflow in the system, one is for the orchestration of the scientific applications (the chain of the data processing) and the other is for the orchestration of the image composition and interactive tools. For the image composition and interactive tools we use client coordinated service chaining. In the simplest case, service chaining is fully transparent to the client who defines and controls the order of execution for individual services in a chain. The client must have prior knowledge of the different service types to be used in the chain.

In this document we first give the definitions for the components of the architecture. In the following chapter, we give the definitions about the scientific applications in a specific domain to which proposed architecture is applied to. In the last Chapter, we summarize the challenges and the proposed solutions.



Scientific Data - stored, transferred, converted, processed and displayed

Figure 1: Sci-Data Web Service Grid Architecture

2. Definitions of the Sci-Data Grid Components

2.1. Interactive Portal

Interactive Portal provides users with some capabilities to interact with the remotely located scientific data and services. Data is interpreted and displayed depending on the given parameters by the user. Data can be raw in the form of GML (Geographic Markup Language), CML (Chemical Markup Language), byte array or plain text file in tabular form. Data can also be in the processed form such as image, and they might be coming from WMS [8, 9] or JChemPaint [13].

Interactive Portal provides tools and environment to visualize, manipulate and analyze scientific data. The nature of the scientific applications requires seamless integration and sharing of scientific data from a variety of providers and servers. Interactive Portal mainly serves for the end users and decision makers. It has several capabilities for the decision makers to access and interpret geo-data seamlessly. GIS Portal is built up with the various technologies; among these are Java, Java Servlet and Java Server Pages (JSP), Java-Script, CSS and Web Services.

2.2. Data Source

Data Sources are scientific data providers. They provide first hand data such as features in GML, maps in images, and chemical data in CML. Data are mostly kept in Data Base and provided by the corresponding data provider servers such as WFS (Web Feature Server) [10], WMS (Web Map Server) and JChemPaint. These data are unprocessed data. They might need to be processed depending on the scientific application needs and Interactive Portal users by the Data Processing servers orchestrated by the workflow engine and coordinating with the Interactive Portal. Global Positioning System (Gps), earthquake fault data, seismic data and Interferometric Synthetic Aperture Radar (InSAR) data are some sample data sources in GIS domain. Atoms data, elements and molecules data are some sample data sources in Chemistry domain.

2.3. Data Plotting

We can also call it data interpretation and information extraction server for the sake of generality. This server not just plots the scientific data but also draw graphs, bar-charts, pie-charts and some other outputs as tables or images showing data characteristics and statistics.

We use Dislin [3] scientific data plotting libraries for plotting scientific geospatial data returned from Data Processing services. Dislin is a plotting library containing subroutines and functions for displaying data graphically as curves, bar graphs, pie-charts, 3-D color plots, surfaces and contours. All these services are wrapped as web services in the proposed system. Data processing services outputs come to Data Plotting services through the Data Conversion server. Each raw data coming from the Data processing servers are preprocessed by the conversion server to convert the data into a common format, VOTable [6, 7].

The Data Plotting services can be divided into two general processing groups. These are data conversion processes and data plotting processes. These conversion routines are different from the ones done at the Data Conversion server. Here, data conversion is done with filtering services.

2.4. Data Processing

Processing services provide operations for processing or transforming data in a manner determined by user-specified parameters. In the Data Processing cloud there might be a lot of different types of routines and services orchestrated by a workflow engine in accordance with the application requirements to get some specific output data to be plotted. Outputs of some routines and services might be an input for some others in the cloud. Failures of some applications effect the success of the others or whole system. All the interactions and synchronization between different applications and even services are

accomplished by the workflow engine. As a workflow engine we will be using HPSearch [11] from CGL (Community Grids Laboratory).

2.5. Data Conversion

All the data coming out of Data Processing (the cloud of core scientific applications) under the control of a workflow manager are directed to Data Plotting server by going through the Data Conversion Server. The common data format in our proposed system is VOTables. The VOTable format is an XML representation of the tabular data. The VOTable format's aim is making the online Sci-Data both interoperable and scalable. The interoperability is encouraged through the use of XML standards. When the data encoded in XML, data interpretation and validation will be easier. During the machine to machine communication, data encapsulation in any kind of messages such as SOAP will also be easier. Metadata about the data can also be embedded in to the data. VOTable format enables these. Metadata can be about the owner of the data, application properties of the data, authentication etc. There is also a price for encoding data in XML. When the data encoded in XML, each data element is tagged. The tagging causes overheads in terms of the volume. Increasing in the data volume basically causes increasing in the transfer time and the processing time.

3. Sample Application Areas and Usage

For the more detailed information about the scientific applications to which our proposed architecture is applied, please see the applications link in the project page [15].

GIS - ServoGrid

Patter Informatics (PI): PI (Pattern Informatics) is a technique developed at University of California, Davis for analyzing earthquake seismic records to forecast regions with high future seismic activity.

Virtual California (VC): The VC approach to earthquake forecasting is similar to the computer models used for weather forecasting. The VC model includes 650 segments representing the major fault systems in California, including the San Andreas fault responsible for the 1906 San Francisco earthquake. The simulation takes into account the gradual movement of faults and how they interact with each other.

GeoFEST (Geophysical Finite Element Simulation Tool): GeoFEST is a two- and three-dimensional finite element software package for the modeling of solid stress and strain in geophysical and other continuum domain applications. GeoFEST uses stress-displacement finite elements to model stress and strain due to elastic static response to an earthquake event in the region of the slipping fault, the time-dependent viscoelastic relaxation, and the net effects from a series of earthquakes.

GIS - Home Land Security (Los Alamos National Laboratory)

IEISS (The Interdependent Energy Infrastructure Simulation System): IEISS is a set of software tools that helps analyzing independent energy networks. This project is led by Los Alamos National Laboratory.

The proposed generic architecture will also be applied to Chemistry domain.

Figure 2 shows an application of the proposed Sci-Data Web Service Grid architecture in the GIS domain. Here in this document we do not give enough explanation about the figure. For more information please see [17]. This architecture specifically applies to GIS - ServoGrid applications listed above.

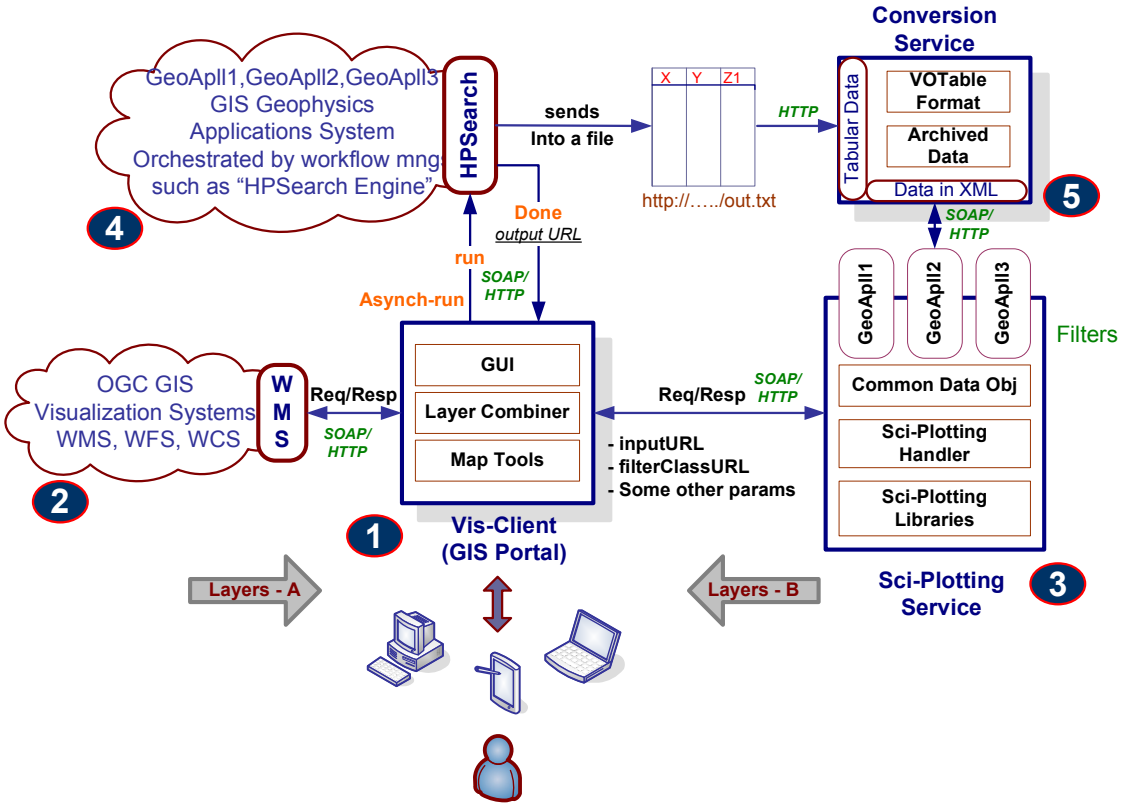


Figure 2: Sci-Data Web Service Grid architecture for the GIS Domain.

4. Challenges

In this section we have listed some possible challenges and corresponding solutions in the tabular form.

Challenges:	*Proposed Solutions:*
Huge Data Transfer	*(NB Streaming [12, 14])*
Huge Data Processing	*(Asynch-Run via WS-Context)*
Common Data Representation	*(XML - VOTables, XDF etc.)*
Distributed interacting Services - Services and data - From different vendors - Run on different platform - Written in different language	*(Web Services)*
Integrating, Inter-relating, Interpreting data from different servers and providers - Raw data - Image data - Tabular data - Statistical data, graphs, charts etc. - Animations	*(Portal)*
Interacting with the remote data on-line	*(Portal)*
Orchestration of Services	*(Workflow Engine for Data Processing)* *(Client Oriented Orchstr at the Portal)*
Generalizing the Characteristics of the Scientific Applications	(….)

For the sake of performance reasons, we will be keeping workload balancing and statefull service approaches in our minds. Since Web Services are stateless naturally, we need to make some extra effort to make the services statefull.

References

[1] Fran Berman, Geoffrey C, Fox, Anthony J. G. Hey., Grid Computing: Making the Global Infrastructure a Reality. John Wiley, 2003.

[2] Foster, I. and Kesselman, C., (eds.) The Grid 2: Blueprint for a new Computing Infrastructure, Morgan Kaufmann (2004).

[3] Dislin project page http://www.mps.mpg.de/dislin/

[4] A Note on Distributed Computing, S. C. Kendall, J. Waldo, A. Wollrath, G. Wyant, A Note on Distributed Computing, Sun Microsystems Technical Report TR-94-29, November 1994. Available from http://research.sun.com/techrep/1994/abstract-29.html.

[5] Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., and Orchard, D. "Web Service Architecture." W3C Working Group Note, 11 February 2004. Available from http://www.w3c.org/TR/ws-arch.

[6] O. Francois, et al., "VOTable Format Definition", Ver-1.1., August 2004.

[7] VOTable web site at http://www.us-vo.org/VOTable/

[8] Sayar A., Pierce M., Fox G. C., Developing GIS Visualization Web Services for Geophysical Applications, ISPRS International Society for Photogrammetry and Remote Sensing Workshop Commission II WG/2 METU, Ankara, Turkey, November, 24-25, 2005.

[9] De La Beaujardière, J. editor, 2002. Web Map Service Implementation Specification. Version 1.1.1, OGC 01-068r3. http://www.opengis.org/techno/specs/01-068r3.pdf

[10] Vretanos, P. A. editor, 2002. Web Feature Service Implementation Specification. Version 1.0.0 OGC 02-058. http://www.opengis.org/techno/specs/02-058.pdf

[11] Gadgil H., Choi J. Y., Engel B., Fox G., Ko S., Pallickara S., Pierce M: Management of Data Streams for a Real Time Flood Simulation Technical Report June 2004.

[12] Message based middleware project at Community Grids Lab, Project Web Site: http://www.naradabrokering.org

[13] Krause S., Willighagen E., Steinback C., JChemPaint – Using the collaborative forces of the Internet to develop a free editor for 2D chemical structures.

[14] Pallickara S. and Fox G., "NaradaBrokering: A Distributed Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids" ACM/IFIP/USENIX International Middleware Conference Middleware-2003, Rio Janeiro, Brazil June 2003.

[15] Sci-Data Web Service Grid Project page http://complexity.ucs.indiana.edu/~asayar/gisgrids/

[16] Bunting, B., Chapman, M., Hurlery, O., Little M., Mischinkinky, J., Newcomer, E., Webber J, and Swenson, K., Web Services Context (WS-Context), available from http://www.arjuna.com/library/specs/ws_caf_1-0/WS-CTX.pdf

[17] Sayar A., Pierce M., Fox G. C., Grid Map Tools and GIS Portal, Technical Report, March 2005. Available from http://complexity.ucs.indiana.edu/~asayar/gisgrids/docs/sci-data-plotting.doc.

Grid Map Tools and GIS Portal

We present an architecture framework to integrate all the GIS services and plotting services with the geo-applications on a portal or GIS Visualization client for the use of end users or decision makers. All components in the system are Web Services based. Interactions of the components are over HTTP or SOAP over HTTP protocols. Since all the services are Web Services based, it is easy to distribute geospatial data and applications across platforms, operating systems, computer languages, etc. They are platform and language neutral. Therefore, it is also going to be easier for application developers to integrate geospatial functionality and data into their custom applications.

Architecture enables overlaying OGC compatible GIS systems' layers with the outputs of the geo-applications. Output of the geo-applications is plotted or superimposed as another layer over layers coming from OGC GIS Visualization systems. There are two generic types of layers which are summarized as below. We call them as Layers-A and Layers-B (see them in Figure 1 as arrows);

- Layers-A are the layers coming from OGC GIS Visualization Systems
    - Coverage Data Layers from WCS [16]
    - Feature Data Layers WFS [17]
    - WMS [15] Layers(Combination of Coverage and Feature Data Layers)
- Layers-B are the layers coming from Sci-Plotting Systems
    - Scientific Data Plotting Layers

Throughout the document we use the term "scientific data". By the scientific data we mean output data of the geo-applications. Scientific data comes from the geo-applications which are orchestrated by the workflow engines. In our proposed architecture we use HPSearch Engine [19] from CGL (Community Grids Lab) [20] as workflow engine but it can be replaced by any Web Service based workflow engine.

There are two types of workflow in the system, one is for the orchestration of the geo-applications and the other is for the orchestration of the layer composition and mapping tools. As it is mentioned above, in our proposed architecture, we use HPSearch Engine as workflow engine for the geo-applications. For the layer composition and mapping tools we use client coordinated service chaining. In the simplest case, service chaining is fully transparent to the client, who defines and controls the order of execution for individual services in a chain. The client must have prior knowledge of the different service types to be used in the chain.

Core plotting services of Sci-Plotting server are provided by Dislin [3] scientific plotting libraries. At the plotting server side, data is located by the URL address given by the visualization client (or GIS portal). Users make the request for a plotting layer by giving the location of the raw data (output of any geo-application) and the name of the filter class in the form of URL for the corresponding geo-application.

Rough data to be plotted might come from a data-base or can be an output of any other sources such as geo-applications orchestrated by any workflow machines. No matter what sources it comes from, it should be converted to a common format to be interpreted by the Sci-Plotting servers. Original data can be any kind such as plain text file or byte streams, but it should be located by their URL addresses. The location information is given to DCS by the Sci-Plotting Server. Upon receiving a request from the client, DCS locate the remote data, gets it and convert it to a common form represented in XML (such as VOTable) and returns it to its client (Sci-Plotting Server).

Interactive GUI and map tools enable users to dynamically interact with the Dislin libraries and GIS visualization services. Each action by the user causes the changes in the parameter values and new request to the corresponding services. Depending on the application and user needs, plotting layers can be shown on another pop-up browser or just overlaid on other layers. All these kinds of service parameters are defined in the requests. Request types and formats for interacting Sci-Plotting Service are defined and explained in Section 3.2.1.

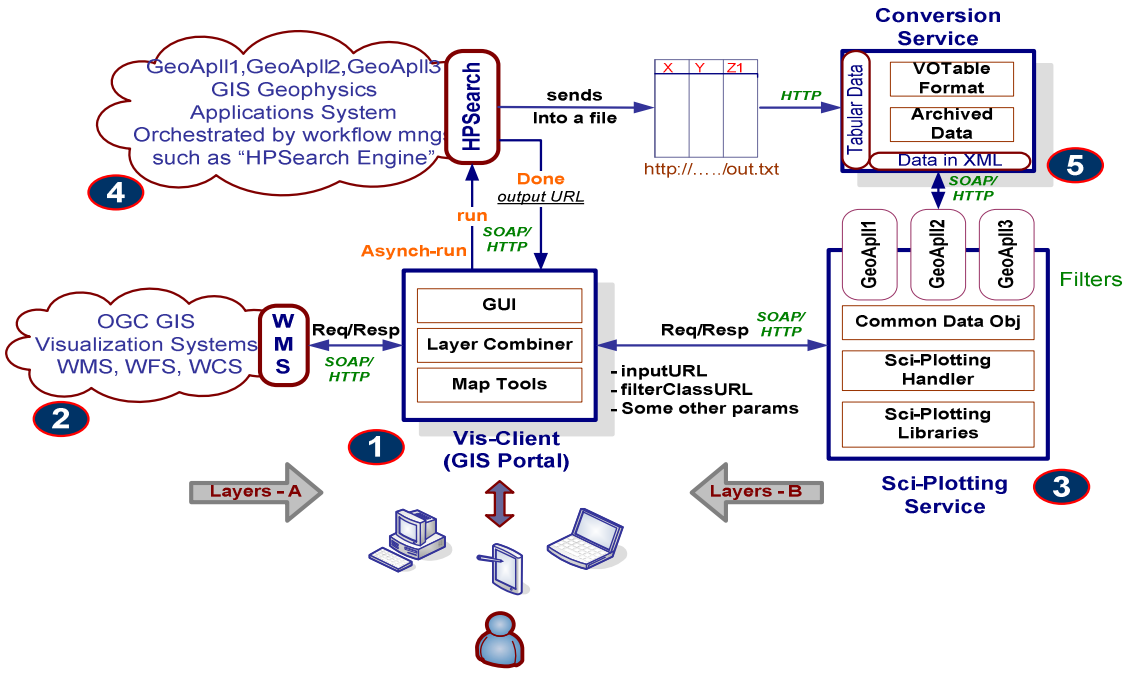


Figure 1: Integrating Sci-Plotting Service into GIS Visualization System

The proposed system illustrated in Figure 1 is composed of four different components. These components are numbered as dark blue circles in Figure 1 and listed below;

- 1. WMS Client or GIS Portal Services
- 2. GIS Services [14]

- 3. Scientific Plotting (Sci-Plotting) Services
- 4. Scientific Geophysics Applications

The following chapters give more details about these components and their interactions.

1. WMS Client – GIS Portal

WMS Client is not just for displaying maps coming from WMS anymore. Rather, it provides some additional capabilities as a GIS Portal. We will be using WMS client and GIS portal terms interchangeably throughout the document. We first start to implement visualization end point as browser based WMS Client. Later we have started to improve its capabilities for being a GIS Portal that combines different kinds of layers from different sources, enables superimposing data plotting layers, interacting and invoking some geophysics applications through the intelligent workflow management tools and software such as HPSearch Engine [19] from CGL. In some contexts, using the term WMS Client is better such as interacting with WMS and displaying the GIS map after receiving the response from the WMS. In all other contexts, using the term GIS Poral makes much more sense such as integrating and orchestrating all the components involving in the GIS systems, integrating third party geop-applications through the workflow engines, interacting and directing the workflow engine, providing signing in or out capabilities, session handling etc.

GIS Portal enables the user end points to request layers from two different kinds of layer providers, WMS and Sci-Plotting servers. Plotting is done over the outputs of the geophysics applications. If these layers, even if they are coming from different servers, are in the same bbox then they can be overlaid together. Please see the Figure 2 for the sample output showing two layers overlaid. In this figure, one layer is coming from NASA [4] OneEarth WMS through our implementation of proxy-master WMS [23] and the other is coming from the Sci-Plotting Server from CGL (Community Grids Lab.). By overlaying different layers from different kinds of servers, decision makers and end users will be obtaining more informatory and explanatory maps. In that sense, WMS Clients can also be considered as GIS Portals that enable user and data-map interactions giving control capabilities to users via some mapping tools. Among these are zooming in, zooming out, panning, getting further information about the displayed data on the map, changing the size of the map and bbox values, selecting prepared regions from the list to display data etc.

GIS Portal interconnects the GIS Visualization Services (WMS and WFS) with the GIS scientific applications and Sci-Plotting services in accordance with the specific purposes of the geo-applications. GIS Portal achieves this task through the user oriented orchestration of the layer composition and mapping tools.

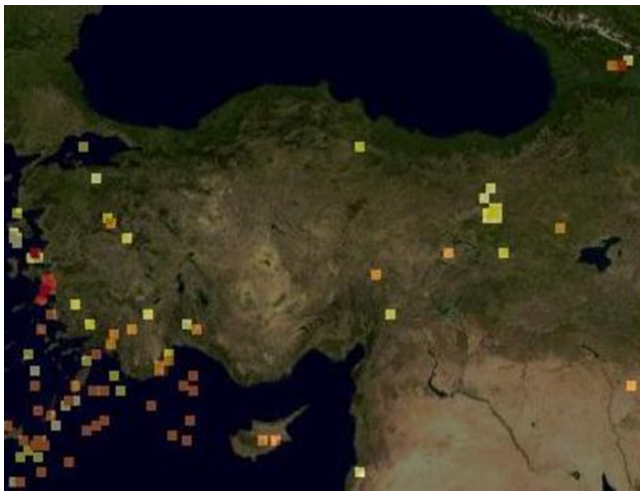


Figure 2: Sample set of layers superimposed.

GIS Portal basically serves for the GIS end users and decision makers. It has several capabilities for the decision makers to access and interpret geo-data seamlessly. GIS Portal is built up with the various technologies; among these are Java, Java Servlet and Java Server Pages (JSP), Java-Script, CSS and Web Services.

2. GIS Visualization Services

Our implementation of Visualization Services is based on the standard specifications defined by Open GIS Consortium (OGC) [13]. OGC defines a number of standards (both for data models and for online services) that have been widely adopted in the Geographical Information System (GIS) community. GIS introduces methods and environments to visualize, manipulate, and analyze geospatial data. The nature of the geographical applications requires seamless integration and sharing of spatial data from a variety of providers. Interoperability of services across organizations and providers is a main goal for GIS and also Grid computing [1, 2].

GIS services, such as defined by the OGC, are part of a larger effort to build distributed systems around the principles of Service Oriented Architectures (SOA). Such systems unify distributed services through a message-oriented architecture, allowing loose coupling, scalability, fault tolerance, and cross-organizational service collections [5]. Web Service standards [6] are a common implementation of SOA ideals, and Grid computing has converging requirements [1, 2]. By implementing Web Service versions of GIS services, we can integrate them directly with scientific application grids [9, 10, 11].

GIS services can be grouped into three different categories; these are data services, processing services and registry, or catalog services [12]. Data services are tightly coupled with specific data sets and offer access to customized portions of the data.

Processing services provide operations for processing or transforming data in a manner determined by user-specified parameters. Registry or catalog services allow users and applications to classify, maintain, register, describe, search and access information about Web Services. In our development of GIS Web Services for the geophysical applications, we use WFS as data services, IS (Information Service) [8] as catalog-registry services and WMS as processing services. For the architecture details see the [23].

We have implemented WMS and WFS services according to OGC standards specifications and as Web Services [14]. IS service is not totally OGC compatible but can be considered as an extension of OGC standards and all its services are implemented as Web Services. Application communication and message transfers between these services are done through the SOAP over HTTP protocol. Request and response instances are XML based and wrapped into the SOAP envelope. SOAP [7] is an XML based message protocol for exchanging the information in a distributed environment, providing standard packaging structure for transporting XML documents over a variety of network transport protocols. SOAP is made up of three different parts. These are the envelope, the encoding rules and the Remote Procedure Call (RPC) convention. SOAP can be used in combination with some other protocols such as HTTP. Our OGC compatible Web Services use SOAP over HTTP.

Even if all the GIS Services involved in our system defined by OGC, they do not all provide their services over HTTP protocol. In the WMS case, some WMSs provide their layers over HTTP protocol and some provide over SOAP protocol. In order to be able to integrate and superimpose different layers from different types of OGC compatible WMSs, we introduced the bridging-master WMS services. It is also OGC compatible WMS but with extended capabilities to serve both kinds of requests coming over SOAP or HTTP protocols. We do that by introducing two requests accepting channels, one is for HTTP requests and the other is for SOAP requests. One channel serves the requests through the Servlets and the other channel serves the requests through the Web Services implementation. All channels use same mapping routines to serve the clients. To be able to invoke SOAP channels, the user should create client-stubs before run-time by using maters WMS server's publicly available service description file (WSDL-Web Service Description Language). After users or external remote applications invoke one of these channels, the server gets the requests and parses it and creates common "Request Object" that the following processing routines can understand. The common request object is created by the "Request Handler" routine. Different channels have a different Request Handler to create common Request Object which extends from abstract Request class. After map creating routines finish their tasks, responses are directed to corresponding channels. For example if the request is done over SOAP, the response will be attached to a SOAP message and returned back to the requestor. If it is done over HTTP then response is written back as byte streams to the request channel.

3. Sci-Plotting Services

3.1. Architecture

In our implementation of Sci-Plotting services, we use Dislin [3] scientific data plotting libraries for plotting scientific geospatial data returned from SERVOGrid applications such as Pattern Informatics, GeoFest and Virtual California. Dislin is a plotting library containing subroutines and functions for displaying data graphically as curves, bar graphs, pie-charts, 3-D colour plots, surfaces and contours. All these services are wrapped as web services in the Sci-Plotting server and integrated into the general visualization system as illustrated in Figure 1.

The Sci-plotting services can be divided into two general processing groups. These are data conversion processes and data plotting processes. Data conversion is done with filtering services (see Section 3.2) and data plotting is done with the service stack and core scientific libraries. Before starting plotting raw data, data should be converted to a common data object, PlotData. Raw data is in the column format and located on an HTTP address which can be accessed by using HTTP protocol. It is actually a plain text file. After PlotData Object is created, the service stack starts plotting procedures by interacting with the scientific data plotting routines through the JAVA Native libraries. Since the service stack is written in JAVA and the core data plotting routines are written in C, service stack routines should use native libraries to be able to interact with the core plotting routines.

Before Sci-Plotting services' data filtering, data is converted to a structured XML form on its way to Sci-Plotting services. This conversion is achieved at the Data Conversion Server (DCS) (see Section 5). Conversion routines are wrapped as web services and easy to integrate into any applications not necessarily GIS applications. It might be physical science, chemistry science or math related science applications that need to use some tabular data. Since its web services based, it can register itself to registry services and can be discovered by a Sci-Plotting server easily.

The aim of the Sci-Plotting services is basically producing plot layers by using generic PlotData. We illustrated generic architecture in Figure 3.

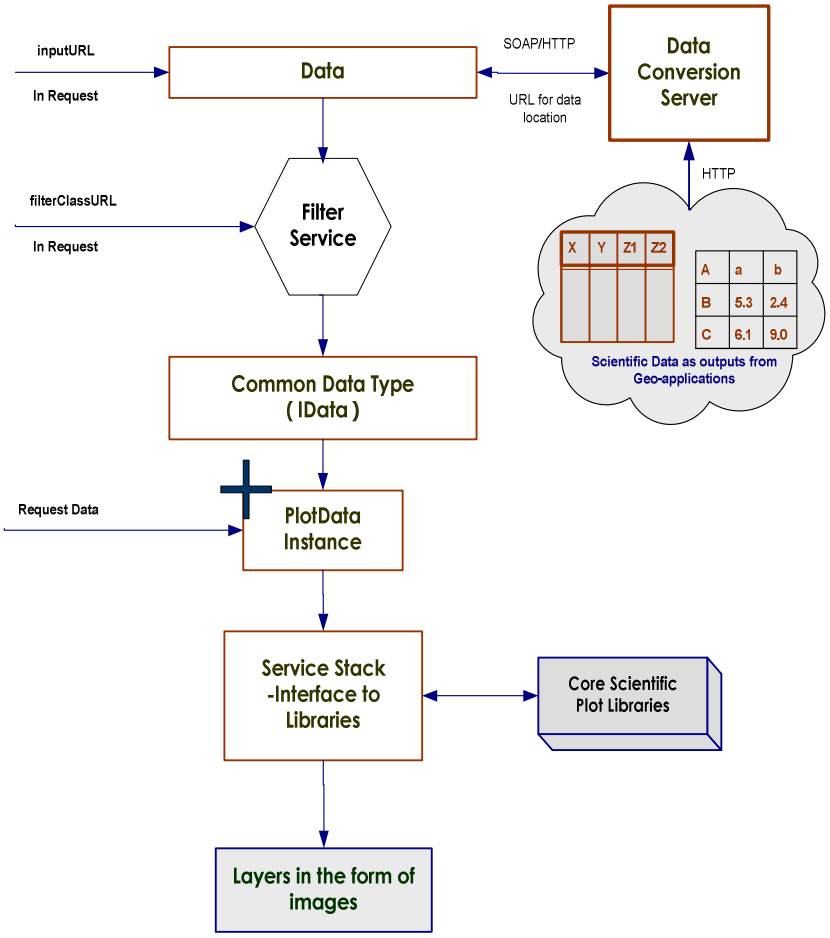


Figure 3: Sci-Plotting Service Architecture. Chapter 3.2 and 3.3 explains the architecture in more detail.

As it is mentioned before, core plotting services are Dislin libraries. Dislin is originally written in Fortran and C but they have also their corresponding native libraries. Native libraries allow us call codes written in any programming language from a program written in the Java language by declaring a native Java method, loading the library that contains the native code, and then calling the native method. We embedded Dislin libraries into our plotting Web Services which is written in Java by using Java Native Interfaces [18] and native methods. In order to install it into our whole system we needed to publish the plotting services as Web Services. Therefore, we first created service description files (WSDL) and publish them by using java axis web services container.

Service stack is composed of native method declarations. All the declared native methods have their own corresponding native implementation routines in the loaded native libraries. Routines can be accessed from any java class through the ServiceStack class. You can see the shrink version of the ServiceStack class as below. Since all the methods are declared as static, class is assumed to be static as well. So, sample contour calls from any java class will be as below;

// before calling ServiceStack routines first import ServiceStack class

ServiceStack.contour(x-array, x-length, y-array, y-length, data-array, data-length, z-level);

```
public class ServiceStack
{
   public ServiceStack ()  {   }
   public static native void contri(float af[], float af1[], float af2[], int i, int ai[], int ai1[], int ai2[]);
   public static native void abs3pt(float f, float f1, float f2, float af[]);
   public static native void contour(float af[], int i, float af1[], int j, float af2[], float f);
   public static native void angle(int i);
   /*
       Native routine correspondences of all the scientific data plotting libraries
   */
   static
   {
      System.loadLibrary("javaNativeCodeLibraries");
   }
}
```

For the sample output of Sci-Plotting services please see the Figure 6. This is for ServoGrid GeoFest application postseis data plotting.

3.2. Filter Service

Since all the ServoGrid geophysics applications are invoked by the end users from the geophysics grid portal, all the Servo Grid geophysics applications give output to be plotted for the end users. Output is composed of columns. Each column represents something which can be understood by the corresponding predefined filtering class. Sci-plotting service receives the output file after having processed by the DCS. Sci-Plotting server sends the URL of the targeted data to DCS, upon request DCS gets the data and conver it to common data format which we call it VOTables. After DCS creates the VOTable format of the output data, data is sent to Sci-Plotting Server. Arriving data goes through the domain specific filtering processes and, at the end, column format data coming from DCS turns into common format data object, IData (see the Section 3.2.2). For more detailed information about the IData, please see the Section 3.2.2. Data are filtered by the filter classes defined by the “filterClassURL” parameter in the Request Object. Sample “filterClassURL” parameter is “http://geoAppl/servo/geofest/postseis”. Since the Filtering Tree is a non-recyclic tree, there is always one path going to any one filtering class such as postseis. Filtering classes are responsible for converting scientific application output data coming from DCS in the form of VOTables into IData. Filtering classes are domain specific classes and they know what the common output format should be and what their corresponding input data are. In order to be able to convert scientific data into IData, filter class should know which column is the x coordinate and which column is the y coordinate and which column is going to be plotted.

Figure 4 shows the general structure of the non-recyclic filter tree. Scientific data as an output of any scientific application comes from DCS upon Sci-Plotting services's request. The address of the location of the data is obtained from the "inputURL" parameter in Request Object. DCS returns VOTables common format version of the data as a response.

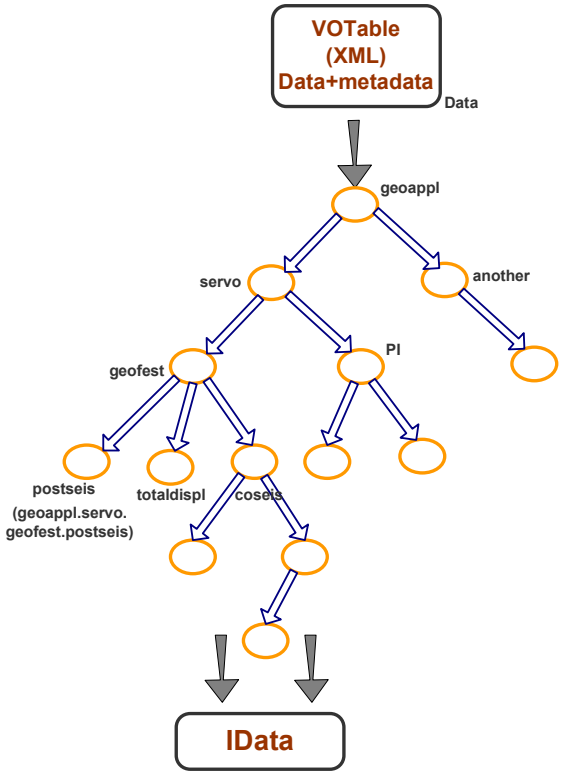


Figure 4: Filter Tree

3.3. Common Data Object for Plotting (PlotData)

For the sake of generality, we have created a generic data object for representing scientific data to be plotted. We call it PlotData (Please see the Figure 5). All the data are turned into PlotData before going through the plotting procedures. Sci-plotting libraries need generic data object and a routine name to be able to plot data. Depending on the data object parameters and routines, plotting output will be different. Data might be gridded data ("regularity"=true) or randomly distributed data ("regularity"=false). "regularity" is a parameter defined in Request Object and sent by Sci-Plotting clients. This parameter defines the characteristics of data and, affects the creation and the formats of coordinates and the data arrays in PlotData.

PlotData is composed of two sub objects, IData Object and Request Object. IData Object is related to Input Data to be overlaid and Request data is related to request object which represents user needs and specific plotting attributes.

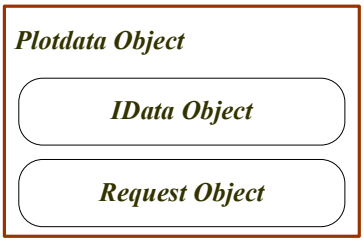


Figure 5: Common Data Object for Plotting

In the following two sections (3.2.1 and 3.2.2), you will see the detailed information about the architecture and possible parameters of the IData and Request Object.

3.3.1. Request Object:

Request Object is created from the values of the request parameters assigned by the remote clients (WMS Client or GIS Portal). Parameters "inputURL" and "filterClassURL" are for creating IData Object. "z-level" array is composed of different elevation values or function values to plot the data in different elevation value ranges. The user can change his "z-level" array depending on the data resolution and other characteristics. The "z-level" should be covering all the data to be plotted to create an explanatory plot layer. The client assigns this array appropriately depending on the geo-application and the characteristics of the output data.

The return type of the Sci-plotting service is a layer in the form of an image. Therefore, clients should assign the "outputType" parameter to an image format such as BMP, PNG or GIFF. These are the Dislin supported image formats. If client is going to superimpose the layer returned from Sci-plotting service over layers obtained from WMS, then, he needs to set "tobeOverlaid" parameter as true. Sometimes, depending on the applications clients need to display returned plot layer in a separate window. In that case, he set the "tobeOverlaid" parameter as false. When the parameter is set to false, the client will have some other parameters that might be assigned to create more explanatory layers. These additional optional parameters are shown below in square braces.

Mandatory parameters in the Request:

String	inputURL	These two parameters are used for creating IData Object. IData Object is a representation of input data.
String	filterClassURL	
float []	z-level	
String	outputType	
boolean	tobeOverlaid	

Optional parameters in the Request:
(in case of the "tobeOverlaid" is set to false)

Boolean	withCoord
String	x-name
String	y-name
Boolean	withTitles
String[]	titles

Incase of the parameter "tobeOverlaid" is set to FALSE. These are additional parameters for creating the graph attributes and appearance.

3.3.2. IData Object:

- Data to be plotted

Currently, our implementation of Sci-Plotting service is generic for 2-dim data plotting. However Dislin libraries can also provide routines for 3-dim plotting. Dislin is our selection of scientific data plotting libraries. In the future we are going to introduce extensions for handling 3-dim plotting. IData object is created by a specific filter class defined by the parameter "filterClassURL" coming from the client. Raw data is read from the column format text file and read into objects. Which column corresponds to which object is defined by the filter class.

For 2-dim data plotting surface is plotted according to x and y coordinate values and data corresponding to these coordinate values. "x", "y" and "data" arrays are created from the raw input data. All the geo-applications sent out their results as plain text file. Filter classes define which column will be assigned to which coordinates and data arrays to convert column data into Java class objects that plotting routines can understand.

Data regularity is an important property for creating coordinate values and plot data, and selecting the appropriate routine to plot. The parameter defining data regularity is called "regularity" and defined in Request Object and used for creating IData object. If the data is regular, filter classes calculate the "xsteps" and "ysteps" parameters. These parameters define the difference in length between successive x coordinate and y coordinate values respectively. IData object also needs minimum and maximum values of the x and y coordinates. These parameters are "xmin", "xmax", "ymin" and "ymax".

float []	x
float []	y
float []	data
boolean	regularity

We assume 2-dim data will be plotted on 2-dim coordinates. x and y are coordinate values.

float xstep
float ystep
float xmin
float xmax
float ymin
float ymax

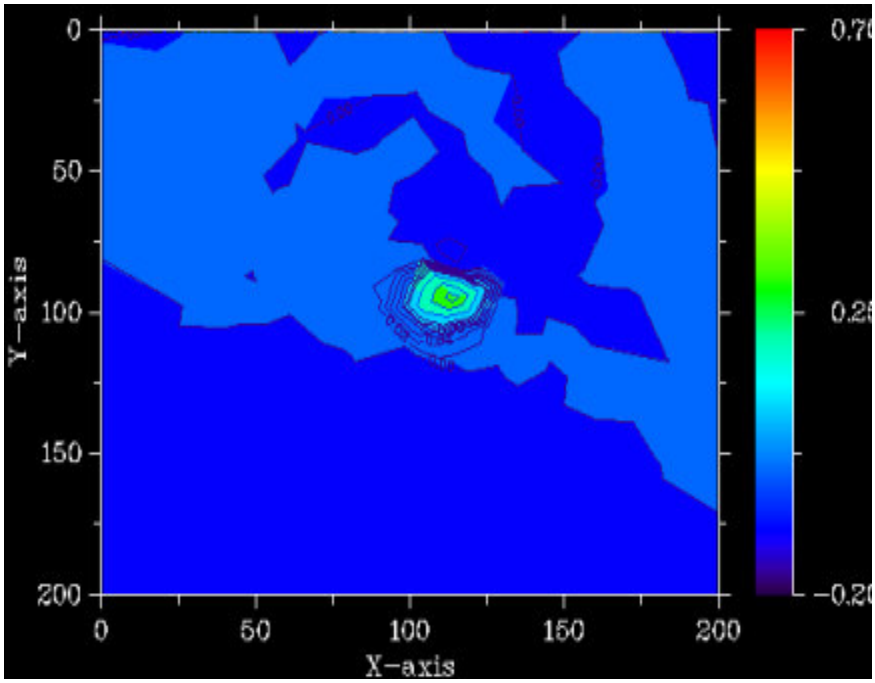


Figure 6: Sample output from Sci-Plotting services. It is GeoFest application from ServoGrid FilterClassURL is http://geoappl.servo.geofest.postseis

4. Scientific Geophysics Applications

In Figure 1, geo-applications are shown as a cloud numbered as 4. There might be a lot of applications involved, standalone or interactively. Outputs of some geo-applications might be input to some others. Failures of some applications effect the success of the others or whole system. All the interactions and synchronization between different applications and even services are accomplished by the workflow engine. We have been applying our proposed architecture framework over geophysics applications in the SERVOGrid project. SERVOGrid builds a distributed computing infrastructure to support earthquake simulation codes. All the applications and services are wrapped as Web Services and involved into the system. Below we list some SERVOGrid applications:

- *Disloc* : handles multiple arbitrarily dipping dislocations (faults) in an elastic half-space. It relies upon geometric fault models.
- *GeoFest* : Three-dimensional viscoelastic finite element model for calculating nodal displacements and tractions. It allows for realistic fault geometry and characteristics, material properties, and body forces. It relies upon fault models with geometric and material properties.
- *Virtual California*: Program to simulate interactions between vertical strike-slip faults using an elastic layer over a viscoelastic half-space. It relies upon fault and fault friction models.
- *RDAHMM*: Time series analysis program based on Hidden Markov Modeling. It produces feature vectors and probabilities for transitioning from one class to another. It is used to analyze GPS and seismic catalogs.

5. Data Conversion Service

All the data coming from geo-applications and specifically from workflow manager as the output are directed to Sci-Plotting services upon requests from GIS Portals and end users. Before Sci-Plotting service gets the output data, the data is processed and converted to a common geo-format by the Data Conversion Service which we call it VOTable [21, 22] format. The VOTable format is an XML representation of the tabular data.

Data comes from variety of different geo applications to be plotted. Each of these applications output different number of records, columns, variables, and different data structures. Most of the applications outputs thousands or even millions of lines of records. Storing, updating and reusing of the data are also a cumbersome. Some times application requires some restrictions and properties on the data. In that case, there should be some properties and attributes about the archived data to check against these requirements. For example, some data are valid just for some specific time intervals or they might have expiration dates for their validations. Furthermore, data on the net may require authentication for access. The remote data needs to be accessed by using various arbitrarily complex protocols by using the URL syntax, “protocol://location”.

The VOTable format seeks answers and solutions to the problems and challenges explained in the previous paragraph. The main goal is making the online geo-data both interoperable and scalable. The interoperability is encouraged through the use of XML standards. When the data encoded in XML, data interpretation and validation will be easier. During the machine to machine communication, data encapsulation in any kind of messages such as SOAP will also be easier. Metadata about the data can also be embedded in to the data. Metadata can be about the owner of the data, application properties of the data, authentication etc. Everything is not perfect when we used the XML encoded data. There is also a price for it. When the data encoded in XML, each data element is tagged. The tagging causes overheads in terms of volume. Increasing in the data volume basically causes increasing in the transfer time and the processing time.

REFERENCES

[1] Fran Berman, Geoffrey C, Fox, Anthony J. G. Hey., Grid Computing: Making the Global Infrastructure a Reality. John Wiley, 2003.

[2] Foster, I. and Kesselman, C., (eds.) The Grid 2: Blueprint for a new Computing Infrastructure, Morgan Kaufmann (2004).

[3] Dislin project page http://www.mps.mpg.de/dislin/

[4] Project OnEarth at NASA JPL (Jet Propulsion Lab) http://onearth.jpl.nasa.gov/

[5] A Note on Distributed Computing, S. C. Kendall, J. Waldo, A. Wollrath, G. Wyant, A Note on Distributed Computing, Sun Microsystems Technical Report TR-94-29, November 1994. Available from http://research.sun.com/techrep/1994/abstract-29.html.

[6] Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., and Orchard, D. "Web Service Architecture." W3C Working Group Note, 11 February 2004. Available from http://www.w3c.org/TR/ws-arch.

[7] Don Box, David Ehnebuske, Gobal Kakivaya, Andrew Layman, Dave Winer., Simple Object Access Protocol (SOAP) Version 1.1, May 2000.

[8] Aktas M., SERVOGrid Information Services Web Site, http://grids.ucs.indiana.edu/~maktas/fthpis

[9] Aktas M., Aydin G., Donnellan A., Fox G.C, Granat R., Grant L., Lyzenga G., McLeod D., Pallickara S., Parker J., Pierce M., Rundle J., Sayar A., and Tullis T. "iSERVO: Implementing the International Solid Earth Research Virtual Observatory by Integrating Computational Grid and Geographical Information Web Services" Technical Report December 2004, to be published in Special Issue for Beijing ACES Meeting July 2004.

[10] Marlon Pierce, Choonhan Yoon and Geoffrey Fox: Interacting Data Services for Distributed Earthquake Modeling. ACES Workshop at ICCS June 2003 Australia.

[11] Geoffrey Fox, et al, Complexity Computational Environnent (CCE) Architecture. Technical Report available from http://grids.ucs.indiana.edu/ptliupages/publications/CCE%20Architecture.doc

[12] Alameh N., Chaining Geographic Information Web Services, IEEE Internet Computing, Sept-Oct 2003, 22-29.

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

[14] GIS Research at Community Grids Lab, Project Web Site: http://www.crisisgrid.org.

[15] Jeff De La Beaujardiere, OpenGIS Consortium Web Mapping Server Implementation Specification 1.3, OGC Document #04-024, August 2002.

[16] John D. Evans, OGC Web Coverage Service (WCS) Specifications 1.0.0, Document #03-065r6 August 2003

[17] Vretanos, P. (ed.), Web Feature Service Implementation Specification (WFS) 1.0.0, OGC Document #02-058, September 2003.

[18] Java Native Interfaces (JNI) page http://java.sun.com/j2se/1.4.2/docs/guide/jni

[19] Harshawardhan Gadgil, Jin-Yong Choi, Bernie Engel, Geoffrey Fox, Sunghoon Ko, Shrideep Pallickara, Marlon Pierce: Management of Data Streams for a Real Time Flood Simulation Technical Report June 2004

[20] CGL (Community Grids Lab) web site http://communitygrids.iu.edu/index.shtml

[21] O. Francois, et al., "VOTable Format Definition", Ver-1.1., August 2004.

[22] VOTable web site at http://www.us-vo.org/VOTable/

[23] Ahmet Sayar, Marlon Pierce and Geoffrey Fox, Developing GIS Visualization Web Services for Geophysical Applications, ISPRS 2005 Spatial Data Mining Workshop, Ankara, Turkey