Scalability Tests for Map Services and "Geo-Science Grids" Integration Architecture

Ahmet Sayar

Introduction

We basically investigate the problems of distributed map processing and rendering of scientific data and information, and the integration of Map servers with the Geo-Science Grids from the interoperability, scalability and performance points of view. Besides being interoperable through Web Services and standard specifications, Map Services should also have some other quality of services to fulfill Science Grids requirements. These might be summarized as being dynamic and up-to-date, being federate-able and being scalable. Dynamicity, updatability and federate-ability are explained in another document. Here we first explain the brief integration architecture, and then explain the tests to test the scalability.

We will implement the Open Geospatial Consortium's (OGC) Web Map Service (WMS) and Scientific Plotting Web Services (Sci-Plotting) by using Dislin scientific libraries and will integrate them with GIS and other services for data access and information management. This will involve cooperation with other lab research projects in GIS and will be integrated with geophysical applications from the ServoGrid project. The GIS services will be described with capability metadata that will test our proposed framework. Capabilities enable federating GIS services and creating pipelines and workflows to solve specific rendering problems (particularly load balancing of complicated images). We will demonstrate WMS federation and Map-Plotting service integration. Based on this work, we will design experiments that will illustrate capability "meta-querying" and service aggregation and federation into "super-services" described above. We will compare and contrast the performance results obtained from the basic Map Services and the one enhanced with workload management and fault tolerance. We will make our tests via our implementation of smart map tools. Smart map tools allow users to create interactive queries and, analyze, display and animate (such as movies for time series data) the geo-spatial information and the data.

Brief Architecture -Tests will be applied on

Integration is done at the layer level. Therefore, the Geo-Science Grids outputs and Map Servers outputs should be represent-able in layers and synchronized as well. Our OGC compatible Map services provide their data in layers. We also use Google Map servers as Map services by using a middleware. This is explained in another document.

Regarding Geo-Sciences' outputs, we need to convert them into layers to integrate (overlay) them with the Map Services' layers. In order to do that we created Scientific Plotting Web Services. It is based on Dislin plotting libraries. After having wrapped them as Web Services, we enabled these libraries to be used by Science community.

There are three set of layers used to build a map with scientific data and information. Layer order is important to create reasonable and human interpretable maps. Please see the Figure-1. Layer-1 is the

bottom layer created from raster data such as Google Earth and coverage data in image format (comes from Web Map Services). Layer-2 is created from vector data such as state boundaries and rivers (comes from Web Feature Services), and Layer-3 is created from interpreted and processed data coming from simulation output of the Geo-Scientific applications. Layer-3 is created at Sci-Plotting Server.

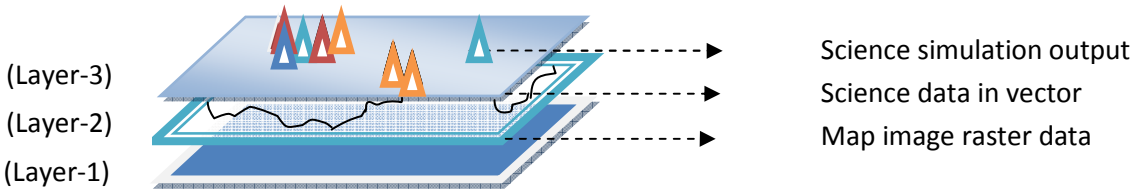


Figure 1: These layers are displayed in the user portal and illustrated in Figure-2 and 3.

Geo-Science grids (or applications) are based on the data (called as features, spatial data or temporal data) related to the earth. They are defined by coordinates, bounding boxes, spatial reference systems, projections, geometry elements and, some other geo-related attributes and elements.

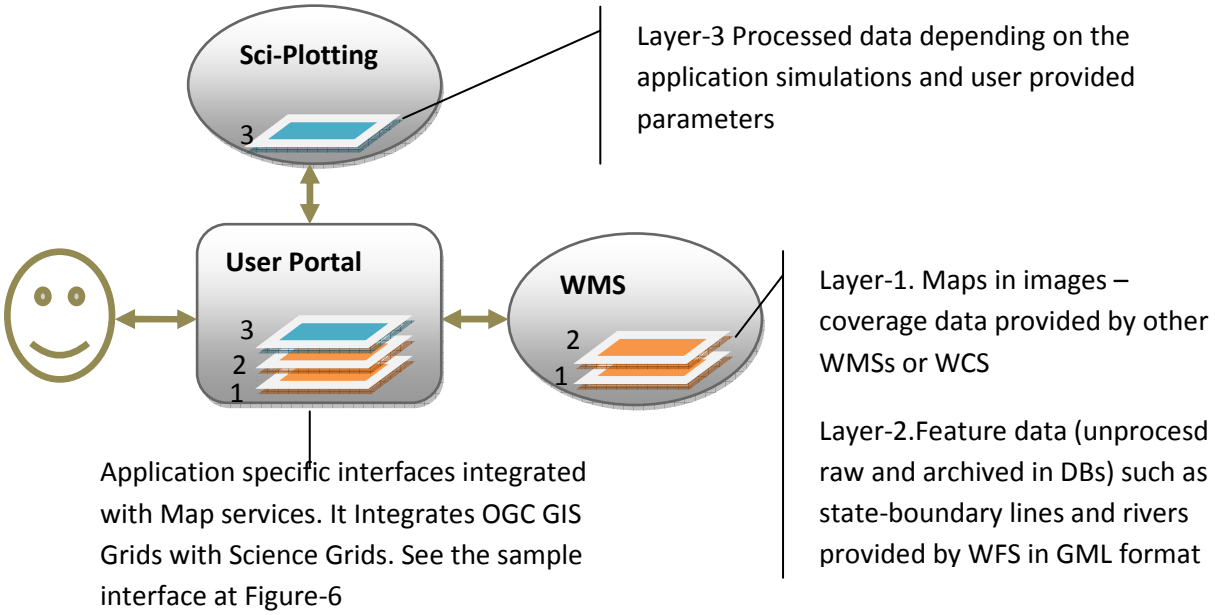


Figure 2: Information comes from the Map Services and Plotting Services in the form of layers in order to get more detailed and comprehensible information.

There are three different groups of layers and three main components in the architecture. Layers are explained above. Regarding the components, there are Map Services, Sci-Plotting Services and User Portal (see the Figure-2) in the system. Layer-1 and 2 come from Map Services and Layer-3 comes from Sci-Plotting services.

Figure-2 shows the architecture roughly without digging into the creation of layers. If you want to see how layers are created in WMS and Sci-Plotting Services, see Figure-3.

Detailed architecture:

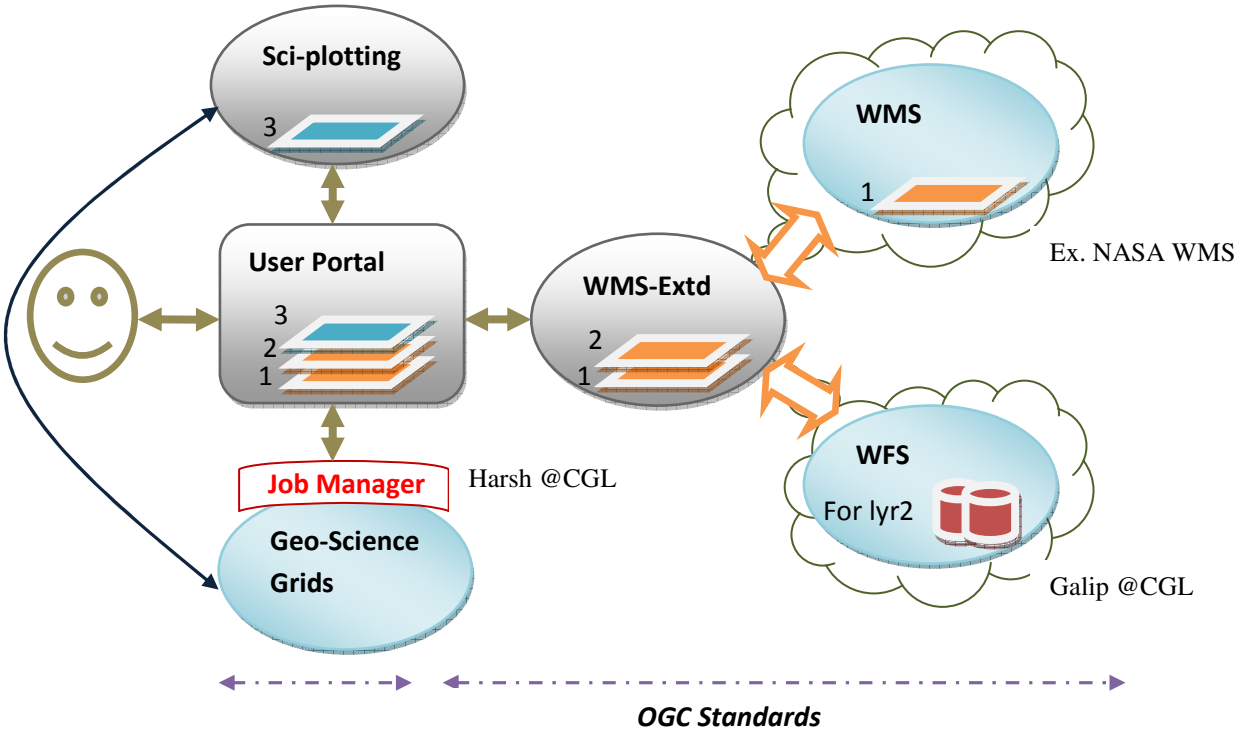


Figure 3: A snapshot from Interactive Decision Support Tools Interface. Sample snapshot is from Pattern Informatics Demo. This is a snapshot from the user portal displayed in Figure-2 and 3.

Web Feature Service (WFS) provide requested geographical information as Geographic Markup Language (GML) feature collections. Data is kept in relational DB and upon request; WFS convert it to GML and returns. WMS interact with a Web Feature Service by submitting database queries encoded in Open Geospatial Consortium Filter Encoding Implementation and in compliance with the Open Geospatial Consortium's Common Query Language.

WMS enables visualizing, manipulating analyzing geospatial data through maps. Map Servers typically compose maps as layers. Layers may come from distributed sources: Web Feature Services provide abstract feature representations that can be converted to images, and other Map Servers may contribute map images. WMSs can be federated and cascaded to create more detailed and comprehensible map images.

WMS-Extd provides all the interfaces functionalities and OGC-interoperability that any other OGC compatible WMS provides. WMS-Extd also provides Google Maps in the image format such as jpeg which can be archived and manipulated depending on the application aims. WMS-Extd has improved performance compared to conventional WMS. It uses load balancing and caching techniques as Google

Map servers do. It also provides properties set before run-time enabling different running modes such as streaming or non-streaming data transfer modes.

Sci-Plotting Services: For the core functionality we use Dislin scientific-data plotting libraries. Dislin is a plotting library containing functions for displaying data graphically as curves, graphs, pie-charts, 3-D color plots, surfaces and contours. All these services are wrapped as Web Services and integrated into the general visualization system as illustrated in Figure 3.

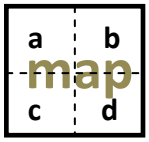
Job Manager and Geo-Science Grids: As a Job Manager we use HPSearch. It is simply a scripting environment for managing distributed workflows. Different projects (IEISS or Pattern Informatics) require different set of parameters for the application users to trigger the job manager. This set of parameters and their order are known earlier. Users enter required parameters through the projects's user interface module deployed in to the user portal. After the application or science grids finish the task, job manager send the output link to the user waiting at the user portal. Then the user requests layer-3 from Sci-Plotting server which is created based on the data located at the link job manager returned.

Approaches to increasing the System's Scalability and Performance:

The system should be able to handle high volume and high rate data transport and processing. Increasing layer number (which means involving more WFS and WMS) or adding more science grid access nodes should not degrade system's overall performance. In order to increase the scalability and performance of the overall system, we plan to apply load balancing, working window and caching techniques, and use topic based streaming data transfer.

Topic based streaming data transfer: We use topic based messaging middleware in order to transfer large data from WFS to WMS.

Load Balancing: We have our own map (WMS and Google) and feature (WFS) services. Therefore we can investigate the problem of distributed rendering of GIS information. We imagine a test scenario in which identical copies of WMS and WFS are spread around the world (we can get accounts around the US, in the UK and Australia (Probably Turkey also). These are basically worker nodes waiting to fulfill requests of a manager WMS. For a specific request, the rendering can be farmed out to worker WMS, which themselves need to find the best WFS.



Four worker nodes' maps merged in the manager WMS. Initial request to manager is cut into four sub requests and returned pieces (a, b, c and d) are merged to create a map as a result to initial request.

Figure 4: Image farming

The WMS workers probably fall into two main subgroups: basemap generators (layer-1) and overlay generators (layer-2). It may be worth distinguishing these for the practical reason that the base map servers (Sunghoon's and Zao's Google Maps servers) require a lot of data in hand (photograph jpegs), whereas the overlay map servers generate their images using data from remote Web Feature Services. Data comes in XML based GML format. Transferring, parsing and rendering GML feature collections are bottleneck of the system performance.

Working Window and Caching: The most recently used bounding box values and corresponding map images are stored by Map Server for the successive requests. Successive requests are handled using the information stored previously. By doing this, sometimes WMS does not need to access the data stored in WFS to create map images. Each user has different session and working window kept in the Map Server.

Below Figure shows the different combinations of successive requests. Boxes 1 and 2 represent requests for the result map. The box-1 is the first and the box-2 is the successive requests.

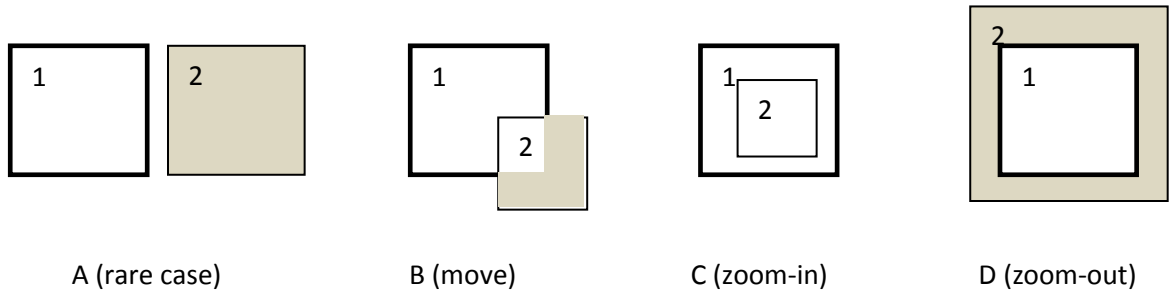


Figure 5: Possible combinations of successive images

In case of box-2, actual requests are done for the grey parts. C represents the full performance gain and A represents the zero performance gain through caching. In C, there is no data transfer between WMS and WFS.

System Tests Based on the Real Geo-Science Applications

We have been applying our framework to three geo-science applications. All the applications are of three set of layers structured. All has similar performance and scalability issues solved using above techniques.

Geo-Science Applications:

1. Pattern Informatics (PI): PI analyzes earthquake seismic records to forecast regions with high future seismic activity. It also identifies the characteristic patterns associated with the shifting of small earthquakes from one location to another through time prior to the occurrence of large earthquakes.

2. Virtual California (VC): Earthquake simulation model for the California. The simulation takes into account the gradual movement of faults and how they interact with each other. It includes 650 segments representing the major fault systems in California, including the San Andreas fault responsible for the 1906 San Francisco earthquake.

3. GeoFEST (Geophysical Finite Element Simulation Tool): 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. It also models static and transient co- and post-seismic earth deformation.

Testing the system for the usability and scalability:

Usability: Making the Geo-Science Applications remotely and seamlessly usable to science community. See the Figure 6.

- **PI:** we run PI code through the integration portal and plot the possibilities of the earthquake happenings in color-coded grid over the previously created seismic and earth map. Seismic data are kept in WFS and queried based on the user provided criteria. We will test the system performance and scalability for the seismic data (GML encoded in XML) transferring, rendering and displaying. We use streaming data transfer through the messaging middleware called NaradaBrokering. We also use NaradaBrokering for the following two Geo-Science applications (VC, GeoFEST). Both o these use seismic data records encoded in GML and stored in WFS.

- **VC:** VC has 2-phase run. In the first phase user runs the application by giving required parameters and get the best costs the result on his screen. If he likes the cost he runs the second-phase with the returned best cost and gets the forecast values. Forecasts values are played in a movie. Base map and seismic records layers are created by the help of load balancing and caching in the Mapping Servers. VC outputs are plotted as Layer-3 at the top. Layer 3 is created at the Sci-Plotting server by using the VC output data.

- **GeoFEST**: Creating successive maps as frames in static images and animating them as movies to see the stress-displacement. Plotting layers (Layer-3) to display co-seismic, post-seismic and complete displacement (the deformations that happen after the elastic response to the earthquake) data. We enable the users to see the visco-elastic relaxation, and the net effects from a series of earthquakes in the form of movies or GIFF-animations.

As an example, let's take the PI application and explain the demo on the Figure-3. User first selects the PI project to work on. At this page he sees the available set of layers for sets of Layer-1 and Layer-2. He also selects the specific regions of earth by giving bounding box values. Layer and bounding-box selection invokes the WMS-Ext. WMS-Ext makes the successive requests to WFSs, Google Map Server or other WMSs to create the map of requested region of the earth and plotting the seismic data over the earth map. In the Figure-6 you can see state boundary feature data coming from WFS through WMS-Extd and Google Maps coming from Google Map Server through WMS-Extd. After having received the

Layer-1 and 2 sets user invokes the *Job Manager* (see in Figure-3) to run the PI simulation to forecast the future earthquake happening over the same region of the earth map. After the PI code running is done *Job Manager* sends the message "DONE" to application portal and user is notified on the screen. User then runs the plotting interface of the Sci-Plotting server to get the third layer to be overlaid over the earth map and seismic data layer. In the Figure-6 you see the color-coded (light-yellow to dark-red, dark-red represents the highest possibilities of the earthquake happenings) gridded layer displaying future earthquake happenings in the California State. You can also note the color scale bar explaining the color codes in the map.

For more details on how to use and run these Geo-Science applications and integrate them with Map Servers through the application portal (Figure-6), please see the user guide at the below link. http://complexity.ucs.indiana.edu/~asayar/gisgrids/docs/usermanual.pdf

Summary of Tests:

After finishing implementation of the approaches mentioned in previous chapter we will test our system. Tests are roughly summarized as below.

Measuring the performance of the basic system

- *We will measure the time to create whole display consisting of three different sets of layers.*

-*We will measure the time to create movies (or GIF animations) from the time series data such as earthquake seismic records. Each frame is a static map consisting of set of layers.*

Standalone server tests: Each server will be tested alone

-Web Map Server: measuring time to create maps (Layers-1 and 2)

-Scientific Plotting Server: measuring time to create plots (Layer-3)

Depending on the data size we will measure end-to end data transfer time. Data is transferred from Feature Server to Map Server.

-*Through messaging middleware (NaradaBrokering)*

-*As SOAP messages and attachments.*

Test Interface- User Portal

Interactive-Smart Map Tools:

Layer-1 and 2 are manipulated through the parts A, C, D and E. Layer-2 is manipulated through part B and utilize the parameters given in part A. Part C is the output screen and enables interactive manipulation of the layers and interactive query of the data rendered in layer-2. If the data rendered in layer-2 is time series data, then part E enables creating movies. Part A enables users to set bbox, map size, specific region to zoom-in, and the layers to be overlaid and project to work with. Part D consists of map tools enabling zoom-in, zoom-out, drag and drop, and data query of the map displayed in Part C. Part B enables users to enter parameters specific to scientific projects. For example for the project "Pattern Informatics", users should enter the parameters "bin-size" and "time-steps". Users can easily move to another project that they want to work by using drop-down list at the top-left corner.

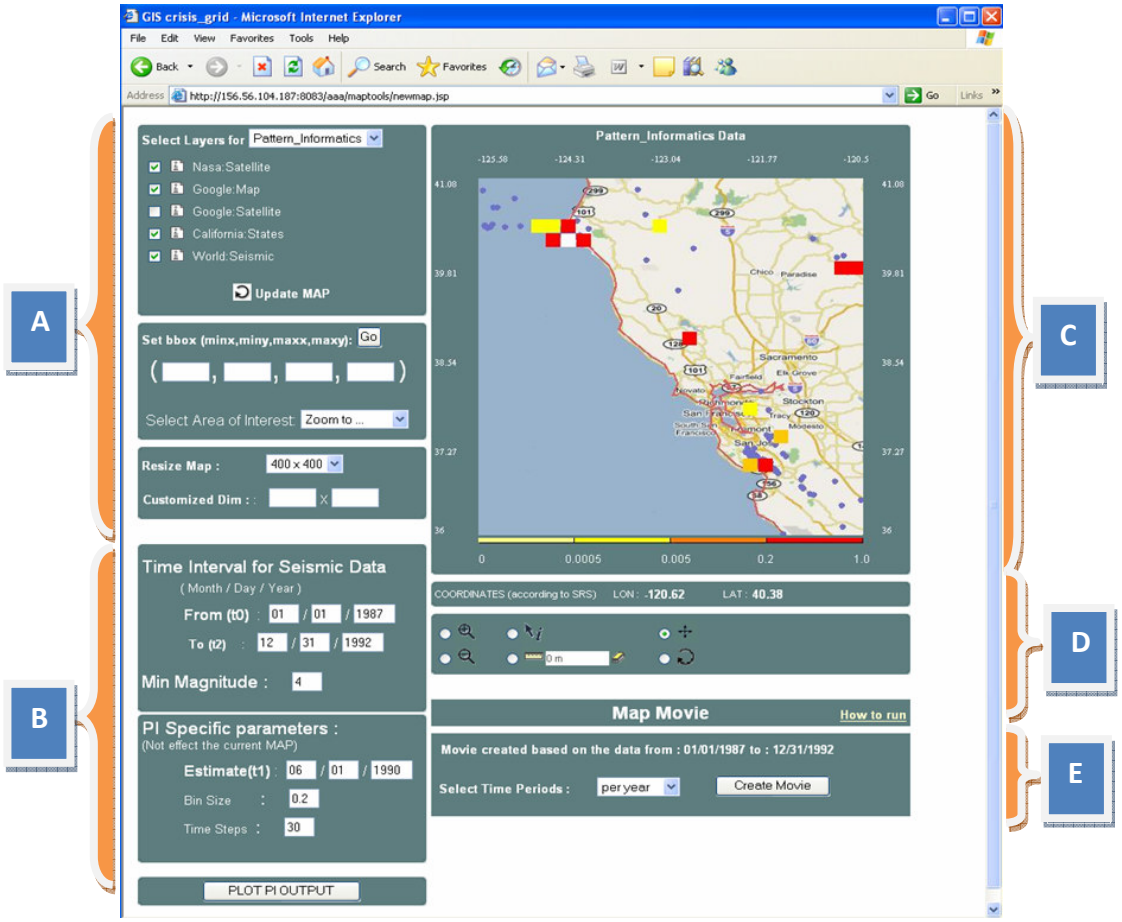


Figure 6: A snapshot from Interactive Decision Support Tools Interface. Sample snapshot is from Pattern Informatics Demo. This is a snapshot from the user portal displayed in Figure-2 and 3.

For more detailed information about using the interfaces for the different Geo-Science projects please see the user guide available at http://complexity.ucs.indiana.edu/~asayar/gisgrids/docs/usermanual.pdf