Grid enabled application visualisation services for oceanographic diagnostics studies

Lakshmi Sastry and Srikanth Nagella
Visualisation Group, CCLRC e-Science Centre
Rutherford Appleton Laboratory
Didcot OX11 0QX
{m.sastry|s.nagella}@rl.ac.uk

Abstract

GODIVA project aims to quantify the ocean overturning circulation that controls climate on long timescales, both from observations and from models. The datasets are so large that static images are inadequate for data analysis, nor can existing visualisation systems support the scalability required. CCLRC e-Science Centre has built scalable application visualisation services using Grid/Web services technologies for real-time data exploration for oceanographic diagnostics. These services can be accessed even from commodity processors using a variety of desktop client front-ends, including Matlab and Java based GUI. This services based architecture with a variety of clients to run on commodity processors actively encourages community wide take up and knowledge dissemination.

1. Introduction

Understanding the relationship between small scale oceanic convection to large scale ocean flows lies at the heart of studies on climatic changes. The ability to carry out heat and fresh water budget studies and thermohaline water mass transformation analyses using assimilated high resolution ocean model data, the dynamic adaptation of this model to an unstructured density mesh and developing the ability to easily visualise such irregularly spaced data in real-time are essential precursors to gain this understanding. These applications are the scientific drivers for the NERC funded GODIVA (Grid for Oceanographic Diagnostics, Interactive Visualisation and Data Analysis) project [1].

This paper describes the progress on the Grid/Web Services for application visualisation that are developed to access oceanography data from third-party data servers on behalf of clients, compute derived quantities and generate visualisation data on the server side dynamically to be sent to the client for local visualisation and interaction.

1.1 Motivation

Ocean models require very high resolution in order to resolve the small, most energetic scales and varying bathymetric depth, particularly near continental shelf edges. Flows on these small scales have an impact on climatically important properties such as poleward heat transport and the locations of sea fluxes, and on biologically important quantities such as net upwelling of nutrients and organic carbon. In addition, the handling of very large quantitative oceanography datasets provided by satellites is required for assimilation methods to study the state of the oceans. The science is multidimensional with complex, nonlinear correlations between the parameters and cannot be understood with static images. Many aspects of the data are also too large to be brought together and analysed by a single research group. Scalability adds an additional complexity for productive data exploration of such very large datasets, necessitating the development of new computing methodologies to produce real-time visualisation, data analysis and just-in-time computing of derived quantities.

The emergence of the Grid/Web Services based computing paradigm with its services for automated co-scheduling, co-allocation and management of resources together with the emergence of local and national high-bandwidth networking offer an opportunity to develop novel problem solving environments (PSE) that cater to the high performance computing, data management and visualisation requirements described above. One of the solutions currently being developed by the authors to the GODIVA data analysis requirements is to re-implement visualisation algorithms as high performance Grid applications in their own right and provide a Web Services based interface which can be published, discovered and invoked with necessary input data. The resulting geometry and/or image data is delivered to the client desktop for local rendering and interaction. In addition we have also implemented Web services that will make third-party data transfer of users data from remote data servers, bypassing the client. Thirdly we are implementing dynamic computation of derived quantities as distributed parallel applications with a Web Services interface so that all the above three can be choreographed into an analysis environment within a client portal interface. A variety of client side user interfaces have been created which make calls to these services to carry out visualisation of oceanographic data. For instance, these data manipulation services are accessed as input by one of the project partners who are developing advanced rendering algorithms for commodity processors [2].

1.2 Background

Computing and data storage on the Grid have been well understood. As a result, large scale computational and data Grids are being setup as infrastructure to support complex and dynamic collaboration between distributed research groups. Grid middleware, the software application programming interfaces to embed secure communication, fault detection and error handling within applications are also being developed to enable the transparent exploitation of the Grid fabric layer. The most prevalent current Grid enabled application model is to submit simulation jobs to a batch processing system, collect data and carry out post processing visualisation for data analysis. Researchers have come to recognise that productive use of time and resources depend on the ability to control these applications interactively while they are running on the Grid. The ability to visualize intermediate results in near real-time at successive time steps is a necessary prerequisite for this control[3]. Also, the quantity of data generated by the simulations of physical phenomena with many hundreds of parameters with complex inter relationships between them demand interactive and flexible visualisation of the parameters and variables of the application to be visualised in near real-time for effective data exploration. This requirement has given rise to current research and development interest focussing on developing interactive visualisation environments to cater to Grid based data analysis.

A majority of current Grid based interactive systems are developed as vertical software for particular applications, with domain specific turnkey visualisation built in as part of the application interface via portal interfaces. The visual images, movies and such are generated on the Grid and delivered to the client-side interface. The focus of such applications is to provide the user with transparent means to query and access Grid resources including data, application and compute power, submit a job on the Grid and retrieve results for post-processing visualisation as shown in Figure 1 below. A variety of client- side interfaces are used to provide interactive control for these tasks [4-6].

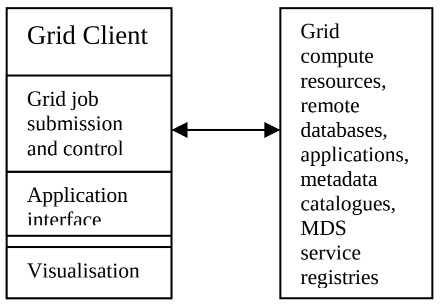

Grid Client
Grid job submission and control
Application interface
Visualisation
Grid compute resources, remote databases, applications, metadata catalogues, MDS service registries

Fig1 Portal based Grid visualisation architecture

Another approach for providing advanced interactive visualisation for Grid enabled applications extends the portal idea to provide more sophisticated interactivity. These include systems that support computational steering for Grid applications via a client-side user interface which also provides the visualisation of application parameters and output as the application executes on the Grid. A range of visualisation solutions are provided for such applications based on the hardware environment available. For instance, raw data from the simulation/application is filtered and mapped on to geometrical objects and high resolution images are produced on the fly using a high-end visualisation engine such as an SGI Infinite Reality system located somewhere on the Grid. The images are then piped through to the client-side steering control user interface for visualisation [7, 8]. High-end visualisation systems such as SGI Visualizer, AVS and VTK are used on the server side to generate the geometry data and the images to be visualised which is transported to client using a variety of data transport protocols. Fig2 below describes a generic architecture of such systems. Where the remote visualisation generator and the desktop visualisation system are the same, proprietary data structures are used to encode visualisation and sent to the client. The above methodologies provide efficient but closed solutions based on proprietary solutions. A variety of complex applications and simulations from a range of scientific areas have been implemented and demonstrated using these minor variations of this architecture.

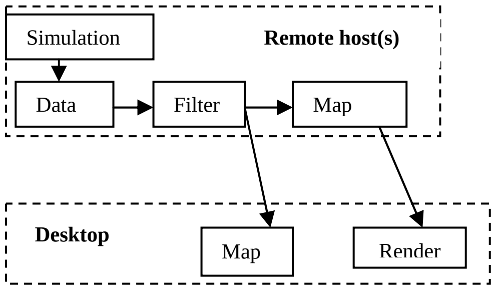


Fig2. Schematic of Grid-enabled dataflow pipeline model (Wood and Brodlie)

An important variation on the above strategy is one where the components of a modular visualisation system such as AVS and IRIS Explorer where compute intensive modules are Grid enabled to support real-time data exploration. This is most useful as intermediate filtering and mapping processes become bottlenecks in the pipeline for certain visualisation techniques or where the data to be handled are too large to handle either for the module or for the hardware system on which it is running. State-of-the-art post-processor visualisation toolkits that address this limitation by distributing individual modules to be executed on the Grid, establish proprietary communication protocols to transfer data between the remote module and the parent running on the client desktop. On the remote system(s), the distributed modules produce visualisation data in proprietary data formats. The toolkit needs to be available both on the server and client sides, making this a closed solution. Problem solving environments such as SCIRun are adopted to exploit Grid computing to support real time data exploration but applications need to be ported to such systems.

Section2 we describe our architecture for supporting Grid enabled visualisation and the requirements that are addressed by the architecture. Section3 describes the services that have been built for GODIVA applications and the clients. This is followed by Conclusions and Future in Section4.

2. Grid application visualisation toolkit architecture

Our approach is to adopt some of the strategies described above but extend these to a generic, open and extensible architecture. It is a bottom up approach with a reimplementation of visualisation algorithms on the Grid, to improve their performance to support real-time visualisation. These codes are then wrapped with a Web/Grid Services based interface so they can be published, discovered and invoked using standard communication protocols and embedded into applications. This allows for applications to choreographed into a workflow, dynamically linking to required services.

The motivation behind our Web and Grid based visualisation services is to provide a generic toolkit that can be used to harness the power of the Grid computing and the dynamic flexibility of the Web/Grid services based architecture and make it available to familiar data analysis environments. Towards achieving this goal, our Grid Application Visualisation Portal toolkit (GAPtk) provides utilities, services, protocols and flexible, configurable high-level application programming interfaces.

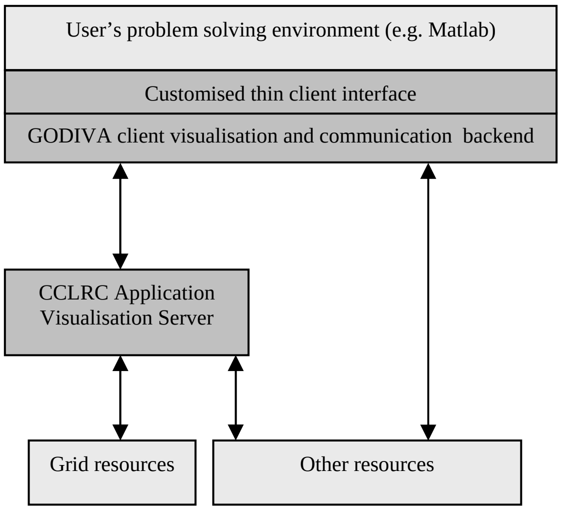
Our visualisation services return visualisation data which is encoded in most generic open data structures which can then be mapped onto a variety of desktop visualisation and problem solving environments for local rendering and interaction. The encoded data structures hold enough information on the nature of the data to allow semantically meaningful behaviour for rendering and for user interaction such as selection, rotation, zoom and inquiries the underlying information on the data (e.g. annotation).

For the GODIVA project, we have built application services that are Grid enabled modules to compute derived quantities on the fly and pipe such data to visualisation services described above. Both application and visualisation services can take URI description to third-party data resources and transfer data to the compute resource without having to create a copy at the client. This allows the desktop users to explore larger amounts of data than what can be handled by local hardware.

2.1 Overvie of GAPtk architecture

Fig3 below provides a schematic diagram of the architecture. At the core of the system is an advanced application visualisation service compatible with the emerging Open Grid Services Architecture, as a mediator between distributed high performance computing and data resources and a range of domain specific application portals using Grid and Web protocols and services.

The software modules in boxes with darker shade of grey are developed by the authors. The customised thin client interface to any generic or domain specific problem solving environments and portals is developed using the GODIVA client backend API by any application programmer. The architecture is designed for portability and extensibility.



User's problem solving environment (e.g. Matlab)
Customised thin client interface
GODIVA client visualisation and communication backend
CCLRC Application Visualisation Server
Grid resources
Other resources

Fig3 Grid enabled data analysis toolkit architecture

2.2 Client-side modules

The services and utilities of GAPtk can be accessed via customised self-contained application portals built using domain specific tools such as Live Access Server and MATLAB using GAPtk clientside API.

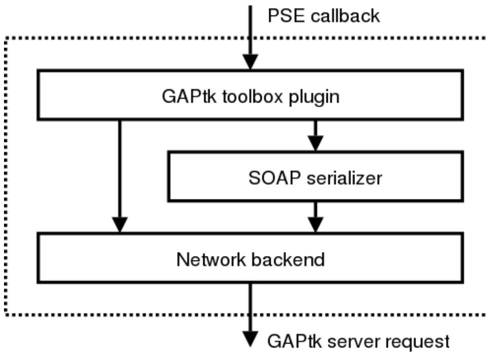
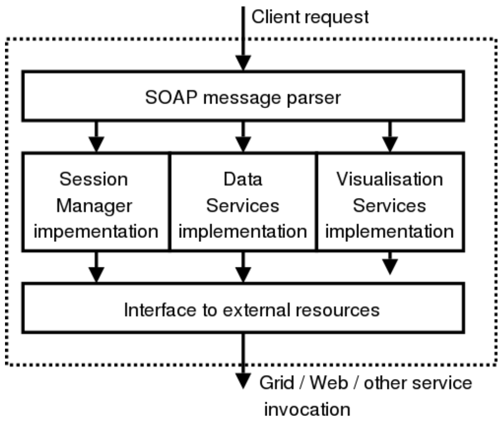


Fig4. Client backend detail

The client backend (Fig.4) is a thin interface designed to translate the client requests into SOAP[9] messages to the server. We currently have three implementations of this part, one using the Java-based Axis toolkit[10], another using a custom C library and the third using gSOAP.

2.3 Serverside details

The server frontend (Fig5) is currently based on the Axis library running as a servelet in the Apache Tomcat container. It's purpose is to map the requests on to the implementation backend.



Client request
SOAP message parser
Session Manager impementation
Data Services implementation
Visualisation Services implementation
Interface to external resources
Grid / Web / other service invocation

Fig5. Server details

Finally, the server-side implementation backend (Figure 3) is responsible for actually carrying out the request. It may do this in various ways, locally on the server for minor queries (e.g. "List the data sets the server has access to"), delegated to specific external data holdings/computational resources, invoke other external Grid or Web services, etc.

The server needs to communicate directly to external resources, for example metadata catalogues. For instance, with the GADS data server [11, 12] for querying the data holding, requesting a sub-sample of data. The server makes use of appropriate data transfer protocols to access the data using the URL from the GADS server. Other services within the server then map the raw/numerical data onto geometry and sends the geometry data to the client that requested the data.

2.4 Application user requirements

The NERC funded GODIVA (Grid for Oceanographic Diagnostics and Interactive Visualisation and data Analysis) project leverages on the above architecture to build a commodity processors based oceanographic data analysis and visualisation application portal for that community.

The use of existing tools for data analysis has automatically lead to the realisation that most of the visualisation techniques required are well established. These are the traditional 2D and 2D vector plots, 2D contours, isosurfaces and volume visualisation for a couple of scenarios. Nevertheless, it also emerged that the physics of the problem is often 4 or more dimensional and complex visualisation scenarios needed to effectively and productively explore the data. As a result what emerged as a key "will be most productive and useful to have" requirement was that far from needing a single step generation of a high quality geometric representation of a huge amount of data, the scientists need the ability to dynamically choreograph complex visualisation scenarios using modular services. This is directly mapped within the GAPtk server, including the flexibility to determine the resolution of the display as well as how they wish to overlay different geometric data to representing different parameters of the problem domain, compare and contrast the effective scales within which each parameter shows variation. Such flexibility for data analysis requires reliable and robust ability from the Grid enabled application visualisation services to respond in near real-time. The data resolution demands adoption of appropriate transfer protocols, formats and strategic just-in-time compute decisions. These requirements are being built in as intelligent services within GAPtk server.

3. Visualisation services

3.1 Isosurface Service

This service generates an isosurface from scalar value file. The isosurface is generated using parallel marching tetrahedron on a Beowulf cluster and the resulting geometry information is stored in a file. Currently the geometry information is written in our specific geometry file format in hdf5. There is a client interface library for reading and writing our geometry file.

3.2 Slice service

This service generates slices from a scalar value field . The slice information is written in our specific geometry file format. There is a client interface library for reading and writing this geometry file format.

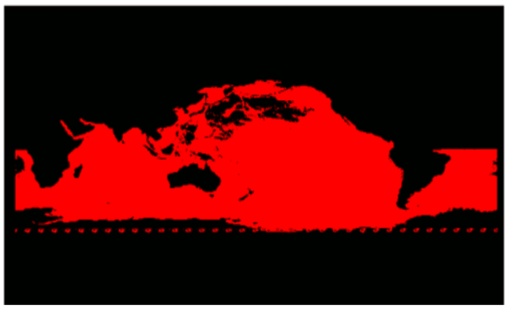


Fig.6 Isosurface of potential temperature generated on the fly

The image above is produced from 600MB of data, where the complete data loop of making initial query and request to the GAD server to getting the data, computing the potential using a Beowulf cluster situated in another building to the visualisation server (but within the same LAN), generating the isosurface and writing this image and sending it back to the client took 2.6 seconds [The computing operates a batch queue system and no special allocation was given].

3.3 Animation service

This service generates an mpeg movie from a time series data. The mpeg is generated in parallel. It takes the geometry information from the time series and generates images in parallel and stitch them to form a mpeg animation.

Performance measurements and improvements are being made to this service at the time of writing this document.

3.4 HardCopy service

This service generates a rendered image in a PDF/EPS/PS format which can be printed without loss of quality. This service takes geometry information and write them into lossless image quality file.

3.5 Data manipulation service

Density calculation service calculates sea water density. The input to this service is a file containing sea water potential temperature and sea water salinity variable data. The calculation is done in parallel Beowulf cluster and an url pointing to the density file is sent back to the user. Similar descriptions apply to the computation of heat flux and surface flux.

In addition, we have also built a spectral transformation service which extracts subsamples of irregular grid data stored in Grib file format and filters and transforms that data to any other required format and coordinate system. The service can take a user provided transform function.

There are several other services, especially those for session management and housekeeping and annotation are currently under development.

4. Conclusions

The most valuable experience we have gained is negotiating firewall policies within and between organisations, the fluidity of the various Grid/Web Services technologies that we are currently using and the need to build simple APIs, define generic data models so that client-side developments can be easier. Also a client-side intelligent module is being developed to determine when the remote Services based data manipulation modules need to be invoked and when these operations can be carried out at the client side itself.

Our aim is to complete the toolkit development with clearly defined APIs so that these can be downloaded and installed at other facilities as part of GODIVA project deliverables.

1. http://www.e-science.clrc.ac.uk/web/projects/godiva
2. J.M. Brooke, J. Marsh, S.Pettifer, and L.Sastry. (2004) The importance of locality in the visual analysis of large datasets, AHM2004.
3. D.Gavaghan, S.Lloyd, D.R.S.Boyd, P.W.Jeffreys, A.Simpson, D.F.MacRandal, L.Sastry, K.Kleese van Dam (2004) Integrative Biology - exploiting e-Science to combat fatal disease. AHM2004.
4. Xu, F, Eres, M.H., Baker, D.J, and Cox, S.J. (2004) Tools and Support for Deploying Applications on the Grid. IEEE SCC 2004 Grid and Utility Computing Track, Shanghai, China, 15 - 18 September 2004.
5. Gabrielle Allen, Kelly Davis, Konstantinos N. Dolkas, Nikolaos D. Doulamis, Tom Goodale, Thilo Kielmann1, André Merzky, Jarek Nabrzyski, Juliusz Pukacki, Thomas Radke, Michael Russell, Ed Seidel, John Shalf and Ian Taylor. (2003) Enabling Applications on the Grid: A GridLab Overview. International Journal of High Performance Computing Applications: Special issue on Grid Computing: Infrastructure and Applications, to be published in August 2003.
6. S. Parker, M. Miller, C. Hansen and C. Johnson. (1998) An integrated problem solving environment: the SCIRun computational steering system. In Hawaii International Conference of System Sciences, pp:147-156.
7. J. Wood, K.W. Brodlie and J. Walton (2003). GViz – Visualisation and Steering for the Grid. AHM2003.
8. J.M.Brooke, P.V.Coveney, J. Harting, S. Jha, S.M. Pickles, R.L. Pinning and A.R.Porter (2003). Computational Steering in Reality Grid. AHM 2003
9. http://www.w3.org/TR/2003/REC-soap12-part1-20030624/
10. http://ws.apache.org/axis/
11. A. Woolf, K. Haines, C. Liu. (2003), A Web Service Model for Climate Data Access on the Grid, Int. J. HPC Applications, 17, pp: 281-295
12. J.D.Blower, K. Haines, C. Liu and A. Woolf, GADS: Using Web Services to access large data sets. All Hands 2003.