Scalable application visualisation services toolkit for problem solving environments

Lakshmi Sastry and Martin Craig
CCLRC e-Science Centre,
Rutherford Appleton Laboratory
Chilton, Didcot, OX11 0QX. UK

Abstract

Much research and development expertise had gone into developing domain specific data analysis, computational steering and visualisation software tools. The use of such tools has become an integral part of modern scientific research with scientists having developed highly customised, robust data analysis applications that form the core of their daily scientific activity. At the same time, advances in experimental techniques, instrumentation and high performance computing are producing ever-larger data sets. The scalability requirements to handle such massive data sets are generally beyond the capabilities of existing data analysis and visualisation systems. The motivation behind our scalable application visualisation services is to provide a generic toolkit that can be used to harness the power of Grid computing and make it available to familiar data analysis environments. The advantage of this approach is that it preserves existing knowledge, familiarity and the appropriateness of domain specific application environments.

1. Introduction

The current computing challenge is to cater to a range of complex applications that aim to process and visualise and gigabytes of data. The emergence of the Grid computing paradigm [1] 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 address the high performance computing, data management and visualisation requirements that these applications demand.

Application toolkits such as Cactus [2] provide a high level of abstraction on top of Grid toolkits such as Globus [3] for programmers, hiding many of the low level issues. PSE tools such as SCIRun [4] provide even higher-level interfaces for applications to be built on. However, these tools require a steep learning curve and also introduce a new development and execution environment for applications. Customising existing applications to new environments is often labour intensive and this is the primary stumbling block for take up. Moreover, to develop such PSEs *ab initio* represents a failure to exploit the vast amount of research and development that had gone into developing currently used software tools, as well as the resources invested by scientists in learning these tools.

This aspect of legacy software is of immediate significance at our Laboratory (CCLRC - Council for the Central Laboratory of the Research Councils, UK), which is one of Europe's largest multidisciplinary research support organisations. It operates several large-scale scientific facilities for the UK research and industrial communities including accelerators, lasers, telescopes, satellites and supercomputers alongside its active participation in scientific research in astronomy, biology, chemistry, environmental science and physics. The CCLRC e-Science Centre is addressing the scalability limitations of existing tools by creating a generic software framework consisting of a backbone of services that exploit Grid computing, and from which legacy applications can improve their scalability. The distinction between our approach and other portal and Grid aware application developments is that ours aims to preserve existing knowledge, familiarity and the appropriateness of domain specific application environments. At the same time, it augments such applications with the powerful Grid computing environment to process and visualise their data and thus facilitate greater take up of Grid technologies.

The architecture is described in Section 2 and is designed to hide the complexities of the Grid computing infrastructure and minimise, if not eliminate, the need for complete redevelopment of application software. The Grid aware Applications Portal toolkit (**GAPtk**) [5] is the realisation of this framework.

In Section 3 we describe an initial set of applications that formed the basis on which generic user requirements have been gathered. The current prototype provides an initial set of Grid enabled Web services for visualisation and data manipulation that are a vital part of any near real-time scientific data exploration. The implementation details of a sample set of services are described in Section 4. The current status of the toolkit and plans for future development are included in the concluding section alongside some observations on the technologies used and the longer-term goals for the use of the software in real world applications.

2. Architecture

The motivation behind GAPtk is to provide a generic toolkit that makes Grid computing methodology available to familiar data analysis environments. Towards achieving this goal, the toolkit will provide utilities, services, and high-level application programming interfaces.

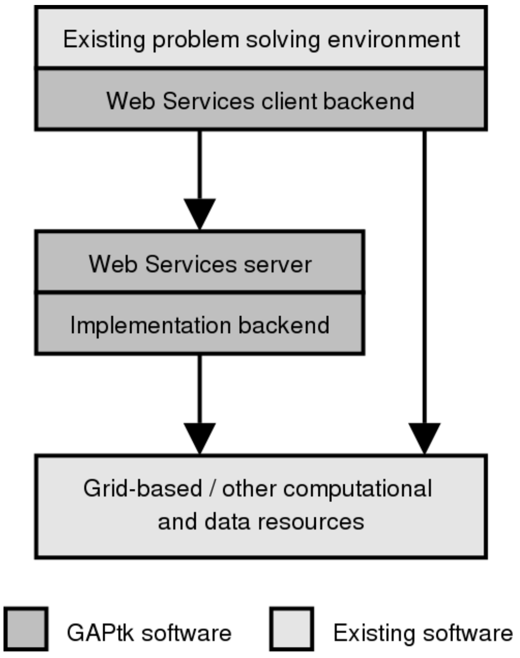
Existing problem solving environment
Web Services client backend
Web Services server
Implementation backend
Grid-based / other computational and data resources
GAPtk software
Existing software

Figure 1: GAPtk architecture

The current prototype is based around Web Services technology [6]. Web Services are request/response-based interactions, which use an XML dialect (typically SOAP [7]) for messaging, and may operate over various communication protocols (though generally HTTP is used). This technology was chosen for a number of reasons, including the simplicity of the protocols that enable thin client side interfaces, the wide availability of support libraries, and the expected convergence of Web and Grid Services. This should provide a migration path in the future to the Open Grid Services Architecture [8](OGSA).

The basic architecture is summarised in Figure 1 and consists of three main parts: a client-side utility consisting of a Web services communication backend and an intelligent rendering and object manipulation library, a server-side Web services front-end, and a server-side implementation backend.

The client backend (Figure 2) is a thin interface designed to translate user requests from application-specific events/callbacks into SOAP messages to the server. We currently have two implementations of this part, one using the Java-based Axis toolkit

[9] and another using a custom C library. In this way we plan to make the services available to a variety of clients, for example a desktop application like MATLAB [10] or a web-based portal such as the Live Access Server [11].

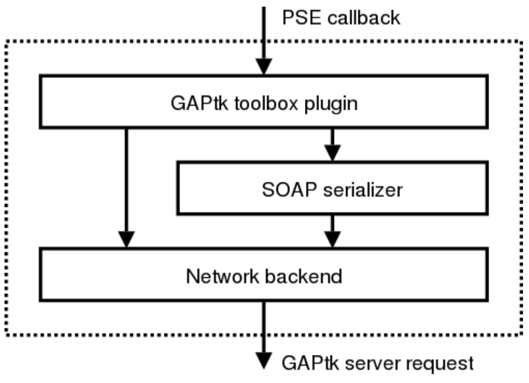

Figure 2: GAPtk client backend detail

The server front-end (Figure 3) is currently based on the Axis library running as a servlet in the Apache Tomcat [12] container. It is a thin layer whose purpose is to map the incoming requests on to the implementation backend.

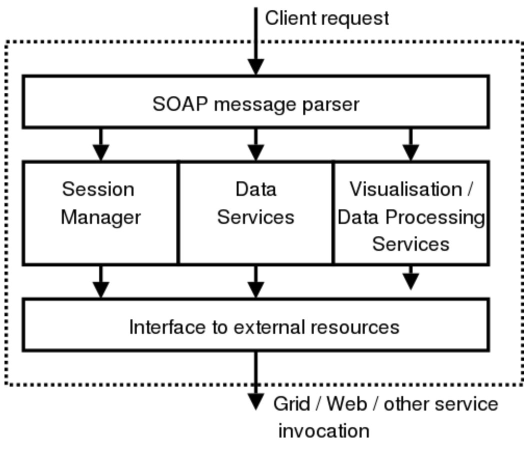
Client request
SOAP message parser
Session Manager
Data Services
Visualisation / Data Processing Services
Interface to external resources
Grid / Web / other service invocation

Figure 3: GAPtk server detail

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") or delegate the request to an external resource, for example another Grid or Web service. It would also be possible for the client to communicate directly with external resources, for example metadata catalogues.

A sample set of Grid enabled Web services within the toolkit

Our current prototype implementation offers two main sets of services, one for data extraction and one for visualisation and data analysis.

The data service allows querying of simple metadata from known data sources. This includes the names of data variables, physical units and dimension ranges. A sample of data can be extracted by specifying a variable, a physical region to extract, and a resolution for the extracted data. In this way large data sets can be easily requested in a more manageable sub-sampled form. Access is provided to multiple data holdings, which may be local collections of files, or remote services. For example, for the GODIVA [13] project (described in Section 3) we have implemented a data holding which interfaces to the GADS [14, 15] data service based at the Reading Environmental Systems Science Centre (ESSC).

The visualisation service allows the execution of basic visualisation tasks, including extraction of isosurfaces and slices, and server side rendering of geometry and animations. Extracted data and the results of visualisation tasks can be downloaded or stored in a session directory on the server where they may be re-used in other calculations. Hence, clients can opt to render geometry data themselves for improved user interaction, or request server-rendered images if they lack this ability. All downloading of data is accomplished using standard HTTP, for efficiency rather than attempting to encode the data into SOAP, which is not an efficient format for large data sizes [16].

3. Applications and user requirements

Key applications

The first production version of the software toolkit is targeted at two key application areas, namely condensed matter physics and environmental sciences. The first of these is represented by the Excitations Visualisation Environment (EVE) [17] project, currently being implemented in collaboration with scientists at the CCLRC ISIS neutron spallation facility. The other is aimed at the UK Oceanographic research community, under the NERC funded GODIVA project. For these two projects, we are also building sample client applications that will make use of the visualisation Web services to demonstrate the key technologies and provide performance and ease of use measures.

Excitations Visualisation Environment

The ISIS facility at the Rutherford Appleton Laboratory is the world's most powerful pulsed spallation neutron source. It provides beams of neutrons and muons that enable scientists to probe the structure and dynamics of condensed matter on scales ranging from the sub-atomic to the macro-molecular. Experimental data sets are typically 0.1-0.2GB per experiment setting, with 1-2GB for a sparse collection of settings This is anticipated to increase to 10-20GB in near future due to new detector instruments.

In most cases, the physics of interest is a function of 4 variables. Visualisation of this data is an important part of analysing the experimental results, and together with data processing (e.g. noise reduction) and simulation, it forms a complex data analysis cycle that needs to be carried out interactively for the scientist to obtain maximum value from the data. A fast, interactive analysis cycle also opens up the possibility of “steering”, i.e. modification of experimental parameters in response to initial results.

EVE is currently being developed as a suite of custom Grid based applications and visualisation services. It aims to leverage the GAPtk architecture to allow advanced visualisation and data analysis at the client-end with a customised portal based on using Matlab^{TM}, since this is the PSE that users are familiar with from the current non-Grid aware system. These data and visualisation services aim to support the physicist through a complete cycle of data analysis in near real-time. This will require an increase in performance over existing tools of at least a factor of 10.

Another key requirement is that the analysis may be performed within collaborative environments, such as the Access Grid [18]. Apart from supporting the scientific collaboration inherent to such large-scale facilities, this will also allow the researcher to guide a local technician to control the experiment in real-time.

GODIVA

The primary scientific aim of GODIVA (Grid for Oceanographic Diagnostics, Interactive Visualisation and data Analysis) is to quantify the ocean overturning circulation that controls climate on long timescales, both from observations and models. This can help to improve the models as well as provide metrics for detecting early signs of climate change hidden within ocean observation.

The Computational science objectives in this project are:

• to make available affordable, familiar and appropriate environments for visualisation and analysis of oceanographic model data.

• To identify and quantify resource requirements and the Grid & Web based technological strategies for deploying a production system on affordable hardware via appropriate portals.

User Requirements

The modular visualisation and data handling requirements of these two widely differing applications have much overlap. The GAPtk architecture was as a result based on detailed user requirement capture exercises carried out in collaboration with scientists working in these application areas. The focus on applications provided extensive insights into existing performance bottlenecks, the scalability requirements for existing and next generation scientific data analysis and exploration and productivity issues.

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 3D vector plots, 2D contours, isosurfaces and volume visualisation. Nevertheless, it also emerged that the data may in some cases be more complex (e.g. 4 or more dimensional) and new visualisation methods are needed to explore the data productively. What has emerged as a key requirement is that instead of a single step generation of a high quality geometric representation of a huge amount of data, the scientists need the ability to dynamically explore small or medium-sized parts of the data. This is directly addressed by the GAPtk server, including the flexibility to determine the resolution of the data sent to the client as well as how they wish to overlay and compare different variables. Such flexibility for data analysis requires reliable and robust ability from the 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.

4. Some observations and Conclusion

The first prototype server and client software is currently in the testing and evaluation phase. First experiences have been mostly positive with regard to the aim of near-real time interaction, and we are currently adding further services and preparing the system for initial user feedback.

A number of important issues have arisen during the implementation of the prototype services. Many of these are related to the need to make the services as generic as possible so that they may be used by clients of widely varying capabilities. For example, when considering what format to use when sending processed data back to the client, a balance needs to be made between efficiency, leveraging robust existing solutions and minimising dependencies. We address this by allowing a choice of data formats, including well-supported existing binary formats (e.g. HDF [19], netCDF [20]) and an XML-based format that is less efficient but removes the need for external dependencies. Internally we use HDF5 as a data storage format for performance, and for the ability to easily read and write subsets of data (this is very difficult with XML because of its free format).

Another particularly important concept that we have yet to address is the idea of workflow, i.e. the chaining together of services to accomplish a complex task. In visualisation this chain may be identified with the visualisation pipeline, familiar from tools such as AVS [21], Iris Explorer [22] and VTK [23]. Our services currently operate as atomic mini-pipelines comprising of data extraction, a single processing step, and data output. Any more complex visualisation must be built up from several web service calls, with resulting loss of efficiency.

An example of this is the extraction of an isosurface of density from oceanographic data and generation of a rendered image file. This requires three services to be invoked, one to calculate density, another to extract the isosurface, and a third to render the image.

It is clear that a solution to this problem would be of great help in improving the efficiency and usefulness of the GAPtk services, and we hope to investigate the possibility of using generic approaches to workflow [24, 25] to accomplish this.

As already described, a framework for collaborative data analysis is also an area of great interest. This would need to allow effective sharing of data and application status between multiple clients and the server. There are solution precedents for these that are tool specific [26], but we believe a generic approach is also possible. One extension that is required to support this is provision for direct communication between the server and client, as opposed to the current request/response mode.

Acknowledgement

The authors wish to acknowledge the support of NERC through the GODIVA Research grant for part of the work reported here.

References

1. I. Foster, C. Kesselman, S. Tuecke (2001) The Anatomy of the Grid: Enabling Scalable Virtual Organizations International J. Supercomputer Applications, 15(3), 2001.
2. G.Allan, T. Dramlitsch, I. Foster, N.T. Karonis, M. Ripeanu, E. Seidel, and B. Toonen (2001) Supporting efficient execution in heterogeneous distributed computing environments with Cactus and Globus, SC2001, November 2001, Denver 1-58113-293-X/01/0011.
3. I. Foster and C. Kesselman. (1999) Globus: A toolkit-based grid architecture in The Grid: Blue print for a future computing infrastructure, Eds. I Foster and C. Kesselman, Morgan Kaufmann Publishers.
4. 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.
5. http://www.e-science.clrc.ac.uk/web/projects/gaptk
6. http://www.w3.org/TR/ws-arch/
7. http://www.w3.org/TR/soap12-part1/
8. I. Foster, C. Kesselman, J. Nick, S. Tuecke (2002) The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Open Grid Service Infrastructure WG, Global Grid Forum, June 22, 2002.
9. http://ws.apache.org/axis/
10. http://www.mathworks.com/
11. http://ferret.pmel.noaa.gov/Ferret/LAS/ferret_LAS.html
12. http://jakarta.apache.org/tomcat/
13. http://www.e-science.clrc.ac.uk/web/projects/godiva
14. 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
15. J.D.Blower, K. Haines, C. Liu and A. Woolf, GADS: Using Web Services to access large data sets. All Hands 2003.
16. K Chiu, M Govindaraju, R Bramley (2002) Investigating the Limits of SOAP Performance for Scientific Computing http://www.extreme.indiana.edu/xgws/papers/soap-hpdc2002/soap-hpdc2002.pdf
17. http://www.e-science.clrc.ac.uk/web/projects/eve
18. http://www.accessgrid.org/
19. http://hdf.ncsa.uiuc.edu/
20. http://www.unidata.ucar.edu/packages/netcdf/
21. http://www.avs.com/
22. http://www.nag.com/Welcome_IEC.html
23. http://public.kitware.com/VTK/
24. http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf
25. http://www.mygrid.info/
26. Jason Wood, Helen Wright, Ken Brodlie. *Collaborative Visualization*, Proceedings of IEEE Visualization 97, pp 253-259, 1997.