
Evolving Dataflow Visualization Environments to Grid Computing

Ken Brodli, Sally Mason, Martin Thompson, Mark Wakley and Jason Wood

School of Computing, University of Leeds, Leeds LS2 9JT, UK

1 Introduction

A major new development in computing in the past few years has been the emergence of Grid computing, where applications can exploit in a secure manner substantial remote computing resources. This offers new opportunities to scientists and engineers. Very large simulations can now be performed, almost routinely, by reserving a distributed set of resources in advance. A key difference from traditional distributed computing is that the resources may span multiple institutions, and so issues of trust become much more significant. In a crisis situation, resources not normally required can be commandeered and dedicated to a crucial simulation or analysis task. In these situations the importance of visualization to gain rapid understanding of the results is well established.

As these new computing technologies emerge, so the challenge for visualization system designers is to evolve existing systems to take advantage of these new technology developments - rather than re-design from scratch on every occasion. The dataflow visualization environment, pioneered over a decade ago, has proved particularly flexible and resilient, and has successfully adapted to the many recent changes in computing.

In this paper, we explain how the dataflow visualization environment can exploit the arrival of Grid computing to provide a desktop interface for remote computational steering. This is by no means the only approach to visualization in Grid computing, and so we begin in section 2 with a brief review of other work. Section 3 traces the development of dataflow systems, showing how they quickly emerged as environments for computational steering as well as data visualization; and how they evolved to encompass collaborative visualization, supporting the activities of geographically dispersed research teams. This sets the context for our work, as we proceed to show how earlier reference models for dataflow visualization can extend to Grid-based visualization. Our research is driven by what can be seen as a typical Grid application: a crisis scenario in which a pollutant escapes from a chemical factory and its dispersion under

different wind directions must be predicted with maximum speed and accuracy in order to plan evacuation. Section 4 describes this application, with section 5 detailing the simulation process we used. Section 6 explains how we evolve the model for dataflow systems to enable computational steering and visualization on the Grid; and in section 7 we describe a demonstrator built from the model, and addressing the pollution application. In section 8, we show how the work can be extended to allow several collaborators to take part in the analysis process.

2 Visualization and Grid Computing

Grid computing is defined as the provision of flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions and resources [8]. Regardless of their heterogeneous nature and geographic location, access to these resources should appear simple and seamless. Only recently, with the advent of high-speed wide area networks, has this ambition become a practical reality. Examples of this can be seen in national-scale computational grid efforts such as the NSF-funded TeraGrid in the US [22], linking nine major computing centres, and the UK National Grid Service [16].

Visualization is key to understanding the ‘tidal wave’ of data that is being generated by scientific applications running on the Grid. In an early paper, Foster *et al* [7] put forward their vision of distance visualization where the computation and visualization are decomposed into steps that may be located on different resources. An application to tomographic reconstruction is described. However they choose to develop their own visualization architecture. This is the approach put forward also by Shalf and Bethel [21]. They argue that existing systems such as VTK, AVS and OpenDX are attractive in their flexibility and extensibility, but were designed primarily for a single machine (or at most a limited number of machines). They go on to claim

But deploying such systems onto the grid requires consideration of new conditions not likely anticipated during their initial design and implementation. A distributed visualization architecture - a framework suitable for component-based, grid-enabled visualization - can best meet these additional design considerations.

Their vision is of a world where distributed, heterogeneous components are available as building blocks to allow application builders to create optimum tools for the task. Our position in this paper is rather different. While the Foster and Shalf-Bethel vision may be a holy grail, there is much we can do right now by a small evolution of existing systems. The reward of our approach is a solution for Grid-based visualization that is available now, and exploits the years of development effort which has been invested in these systems. A particular feature of our approach is that we are able to quite naturally extend from single-user to multi-user working.

Before we introduce our approach, we mention some other instances of using visualization in Grid computing. Computational steering on the Grid has been discussed by Engquist [6]. He uses the VTK software to construct a steering application, but without the ability to collaborate in controlling the simulation. The UK e-Science RealityGrid project has also used VTK, and has developed an API for computational steering [4]. Recently this project has carried out a highly impressive demonstration of Grid-based visualization and computational steering at Supercomputing 2003, in collaboration with the USA TeraGrid - one of the largest ever lattice-Boltzmann simulations was carried out. Data visualization (as opposed to simulation visualization) is also benefitting from Grid computing - Norton and Rockwood [17] describe a progressive approach to view dependent volume visualization. This addresses the important topic of compression - a vital aspect in distributed visualization. Jankun-Kelly *et al* [12] show how existing web-based visualization tools can be deployed through Grid portals - with particular emphasis on spreadsheet-based visualization. Finally, Bethel and Shalf [2] discuss the importance of network efficiency in distributed visualization on the Grid, with experience from Visapult (a parallel volume renderer) and Cactus (a framework for high performance simulation and visualization that has been used for a number of Grid applications).

Our contribution in this paper is to return to the fundamental reference model for dataflow visualization, and show how this model can be extended to include a link to a remote, Grid-based simulation. In this way we provide a blueprint for how to extend the several modular visualization environments which are based on this fundamental model. Therefore in the next section we trace the evolution of dataflow visualization systems, so that we can set the scene for our extension.

3 Dataflow Visualization Environments

Many see the NSF Report of McCormack, de Fanti and Brown [15] as marking the beginning of the modern visualization era. This report sparked the development of a number of dataflow visualization systems, such as AVS [1], IRIS Explorer [26] and IBM Open Visualization Data Explorer [18] - all three of which are still in active use today. These systems were based on a pipeline model of visualization, elegantly presented by Haber and McNabb [11] and shown in simplified form in Figure 1. Data is read in; it is converted to some abstract geometric representation; and this is then rendered as an image on the display.

From the outset, the systems were used both for visualization of data acquired by observation or measurement, and also for visualization of the results of simulations. In the latter case, three models quickly emerged and were nicely characterised by Marshall *et al* [14]. The simplest model is to use visualization as a **post-processing** step: the simulation is executed, the

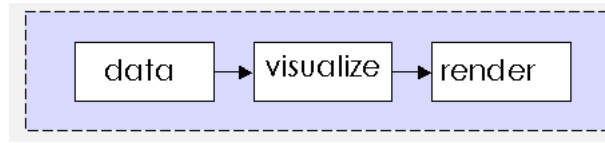


Fig. 1. Basic Visualization Pipeline

results written to storage, and then visualized in a separate step using the pipeline of Figure 1. This has the advantage of allowing the scientist to analyse the results at their own pace, and share the results with others, but it has some major limitations. Errors in simulations cannot be detected until the simulation completes, making mistakes expensive, and there is no opportunity to interact with a simulation on the basis of the intermediate results.

Fortunately a key feature of all the dataflow visualization systems is their extensibility. In addition to the set of modules provided with the system, users can embed their own simulation code as a module. This enables a different model for linking simulation and visualization, termed **tracking** by Marshall *et al.* Essentially the process marked 'data' in Figure 1 is replaced by a 'simulate' process. Now the results can be viewed as they are computed and any errant simulations immediately halted. However even greater flexibility is provided in the third model, termed **steering**, in which control parameters of the simulation can be modified as it proceeds - schematically, this gives a basic pipeline as shown Figure 2. In the pollution application we shall use as demonstrator in this paper, the ability to change wind direction as the simulation executes will be vital.

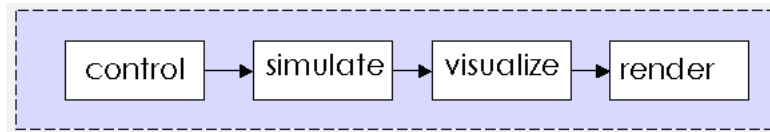


Fig. 2. Steering

The early visualization systems mentioned above (IRIS Explorer, IBM Open Visualization Data Explorer and AVS) can all be used for computational steering in these ways. Moreover a number of newer systems have emerged - such as SCIRun [13, 19] for example - which aim to provide special support for close coupling of simulation and visualization. Toolkits such as VTK [20, 25] can also be used, through the application developer incorporating toolkit components within their simulation code. In all cases however the underlying conceptual model is as in Figure 2.

As the use of the Internet expanded, and more and more research was carried out by geographically separate research teams, so the push came for visualization systems to be collaborative. Again the dataflow systems were

able to evolve, allowing users at different locations to link their pipelines across the Internet - thus enabling exchange of raw data, geometry or images, as well as shared control of parameter settings on modules. This was realised first as COVISA [28], then an extension to IRIS Explorer but now an integral part; and also later as extensions to AVS in the MANICORAL project [5] and in the cAVS project [3]. The model is shown in Figure 3: a collaborative server process links the sessions allowing data from one pipeline to be transmitted to a collaborator's pipeline. Here the data is passed to the collaborator to visualize as they want, but equally the sharing could be programmed to take place further down the pipeline, after the 'visualize' step. This programmability allows the collaborative application to exploit different bandwidths between the locations of the research team. For example, in a low-bandwidth situation, a group might choose to share data at the start of a collaboration, with only parameters being exchanged during the session. Alternatively, with high-bandwidth connections, geometry and image data might be exchanged as the session proceeds.

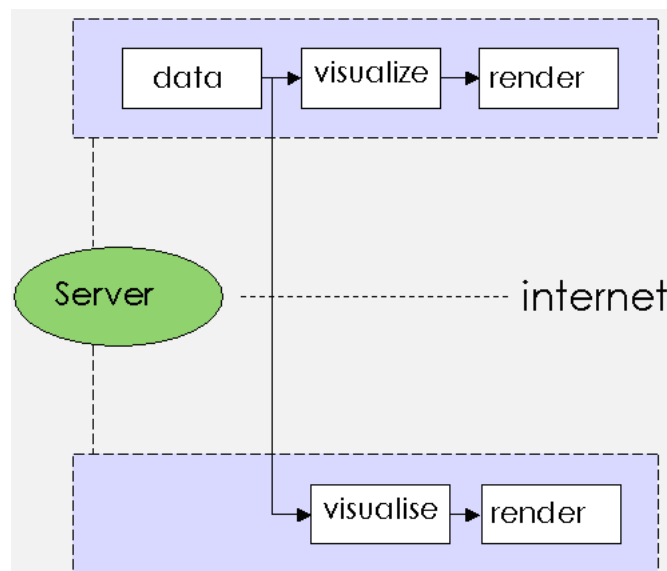


Fig. 3. Collaborative Visualization Pipeline

Our concern in this paper is to evolve the dataflow model one step further, to encompass Grid computing. Rather than immediately describe the extended model in the abstract form of Figures 1 to 3, we shall first motivate the work by describing a typical application of Grid computing where visualization, computational steering and collaboration are all important ingredients. We do this in the next two sections.

4 An Environmental Crisis - Pollution Alert

Imagine this: a dangerous pollutant escapes from a chemical factory - where is it headed? We need to know the answer faster than real-time, so that people can be safely evacuated. This calls for immediate access to powerful Grid compute facilities in order to run a numerical simulation with a desktop interface for the scientist to adjust simulation parameters and to visualize the results. Additional expertise is needed, such as meteorological information concerning likely wind velocities - this is needed instantly, yet the meteorologist will be located elsewhere. Electronic collaboration is absolutely essential to allow the meteorologist to share their knowledge of the problem, a computational steering approach allows them direct control of the relevant simulation parameters - to address questions such as 'what happens if the wind changes to this direction?' - and to share visualization of the resulting effect. Once the scientists have reached an understanding, this needs to be communicated to the political decision maker in a clear and effective manner (the Challenger space shuttle disaster in 1986 is recognised now as a failure of scientists to present an effective visualization to launch decision makers [24]). The decision maker also needs to collaborate over the Internet, with the ability to propose further 'what if' scenarios, but perhaps using a simplified interface to the underlying software. Indeed the actual way the data is visualized will vary according to the different people involved, and the information they need to understand.

The project team will typically have different skills and motivations: the numerical analyst will be more interested in the performance of the algorithms and, for example, the error in the computed solution; the chemist will be interested in the chemical concentrations and details of the physical processes occurring; the decision maker will be interested in regions where the concentrations exceed safe levels and the potential risk to populated areas. The collaborative model allows separate users to analyse a set of data in a completely independent fashion, as shown schematically in Figure 3. Additionally the researchers may work at geographically remote locations. The collaborative model will facilitate the dissemination and discussion of the work by allowing problem parameters, computed data and rendered images to be processed and shared between the researchers. With that motivation in mind, we now describe the simulation involved in our application.

5 Simulating the Dispersal of the Pollutant

Computational models describing the evolution and reaction of chemical species in the atmosphere are an important tool in understanding the formation of hazardous pollutants such as greenhouse gases and acid rain. Such problems are typically modelled with a system of partial differential equations describing the transport of the chemical species through space and the chemical reactions between the species present. The modelled processes can occur

on widely disparate time scales, hence the most effective numerical methods make use of adaptivity both in space and time [23]. The high spatial and temporal accuracy required, coupled with long simulation times, make this an intensive computation. The full model is a major computational challenge, for which Grid computing can provide the resources necessary for its solution. Our simple prototype allows us to explore the sort of problem-solving environment needed to allow interaction with code during the long simulation, to visualize the solution as it evolves and to interactively steer the computation when required. The simulation considered here is a prototype of the full scenario, in which a single component of pollution is emitted from a source and transported through the spatial domain. Figure 4 shows: on the left the scenario of the real-life problem being studied; in the centre, the model used as basis for the simulation, with the computational mesh; and on the right, the visualization of the solution at a particular time-step. The simple visualization shown here, such as may be required by the decision maker, is to select a threshold value for the concentration and render the cells that exceed that threshold.

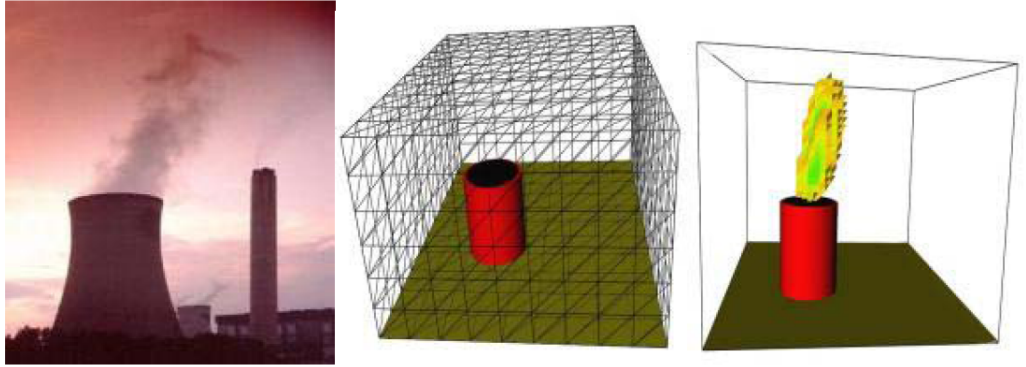


Fig. 4. The Reality and the Model

The mathematical model is an advection equation for a scalar quantity $c(\mathbf{x}, t)$ (the concentration of the pollutant) in a spatial domain Ω with boundary Γ and time domain $[0, T]$

$$\frac{\partial c}{\partial t} + \mathbf{a} \cdot \nabla c = f(\mathbf{x}) \quad (1)$$

where the spatial distribution of the scalar c is driven by an advection velocity $\mathbf{a}(t)$. Note that \mathbf{a} is spatially uniform but can vary in time allowing the convection direction to change. The source function $f(\mathbf{x})$ is used to simulate the production of the scalar c inside the domain, such as from the chimney of the chemical factory in our application. A cell-centred finite volume method

is used to approximate the governing partial differential equation with an unstructured tetrahedral grid representing the spatial domain. The results are returned as the concentration of pollutant within each cell, at each time step of the simulation.

6 Computational Steering in a Grid Environment - Reference Model

Our aim here is to illustrate through the example of the environmental crisis, that the traditional dataflow visualization system is perfectly well suited to Grid Computing. We aim to provide an easy access to important aspects of Grid computing: authentication, via single log-on; resource discovery, via enquiry of Grid information services; and resource access, via remote launching of simulations from the desktop.

We assume the authentication step is performed before commencing the visualization processing, but we need additional modules that will support resource discovery and resource access. The ‘discover’ module queries the Grid information service for available resources, allowing the user to select a specific resource to be used for the simulation. The ‘link’ module receives the detail of the selected resource, and uses that information to initiate a simulation on the Grid and receive results from it. Figure 5 shows a schematic view.

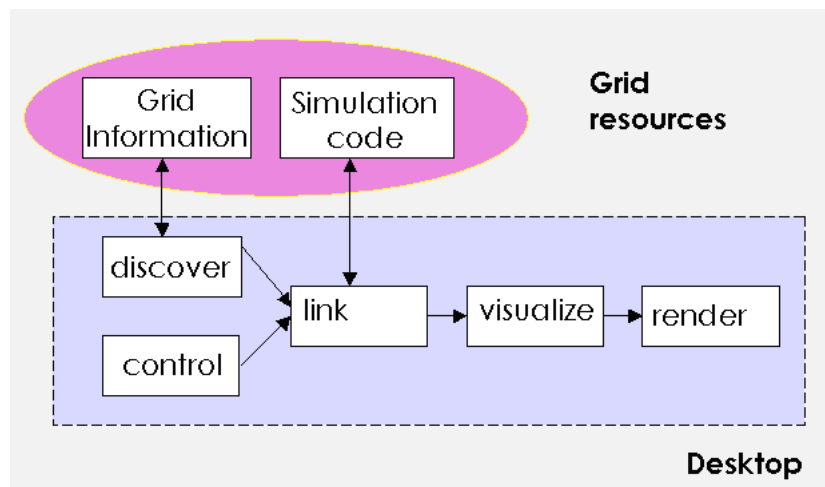


Fig. 5. Grid-based Computational Steering

7 Computational Steering in a Grid Environment - Demonstrator

As a validation of the model, we have implemented a demonstrator that tackles the pollution problem of section 4, using the simulation described in section 5.

Our Grid computing environment was the White Rose Grid [27]. The White Rose Grid is a regional-scale Grid environment that incorporates four major high performance computing resources distributed between the universities of Leeds, Sheffield and York in the UK. It has been constructed to explore the potential for optimising resource usage between institutions and for supporting scientific collaboration between academics and also their industrial partners.

The Grid middleware used within the White Rose Grid is Globus Toolkit v2.4 [9]. Key components of the Globus Toolkit include: the Grid Security Infrastructure (GSI), the Globus Resource Allocation Manager (GRAM) and the Monitoring and Discovery Service (MDS). These have all been used in the demonstrator. It is important to realise that Grid computing is a young and evolving subject, and the concepts and technology have still to mature. We believe the abstract model of section 6 will remain applicable as this evolution occurs, but the Globus technologies used in the demonstrator will be replaced by new developments, as increased focus is placed by the Grid community on Web services.

We have used IRIS Explorer, a long established visualization system, as the basic visualization environment for our demonstrator. Figure 7 shows the system in use.

The GlobusSearch module queries the MDS, the Globus information service. Each Grid resource maintains a catalogue of information that describes various static and dynamic features of that resource. For example, information recorded for a computational resource would include number of CPUs, size of memory and current workload. The Grid Resource Information Service (GRIS) is the Globus component responsible for maintaining this metadata for an individual resource. The Grid Information Index Server (GIIS) pools together the information from each GRIS and thus provides a convenient single point of reference for information about all resources within a single Grid. The ‘GlobusSearch’ module therefore connects to the GIIS, making an LDAP query to discover the details of the Grid resources. The module displays, for each host, the machine name, its operating system, memory size, number of processors and current load. From this information the scientist can select the most appropriate host on which to run the simulation. The host name is output from the module, and wired to the GLSSpawn module which links the desktop to the simulation running on the Grid.

The GLSSpawn module initiates the specified executable on the selected host, using the `globus-job-run` command. In this example, the executable code is transferred from the desktop to the selected host prior to execution, using Globus. The simulation is compiled for IRIX, Solaris and Linux OS’s so any

of these platforms may be targeted. Our experimental Grid offers a ‘fork’ job manager allowing our simulation to start immediately upon submission. Once started, the simulation connects back to the module by means of sockets and then will lie dormant, polling the desktop for a ‘run’ flag. On receipt of this, the simulation is triggered and executes on the Grid, returning results to the GLSSpawn module. The frequency of reporting data from remote host to desktop is controlled by settings on the module interface, as is the frequency with which steering parameters are requested. The simulation can be paused at any time to allow time to make steering decisions, or equally parameters may be changed while it executes which the simulation will pick up at the next update point. The interface panel contains generic parameters such as time stepping, pause/run controls and grid size. An application-specific module is used to steer the simulation, by direct manipulation of a 3D arrow widget that controls the wind velocity.

8 Collaborative Computational Steering - Reference Model and Demonstrator

The final step is to allow a research team to handle the simulation. Here we combine the Grid approach of the previous section with the collaborative visualization approach of Wood *et al* [28] - giving the schematic model of Figure 6.

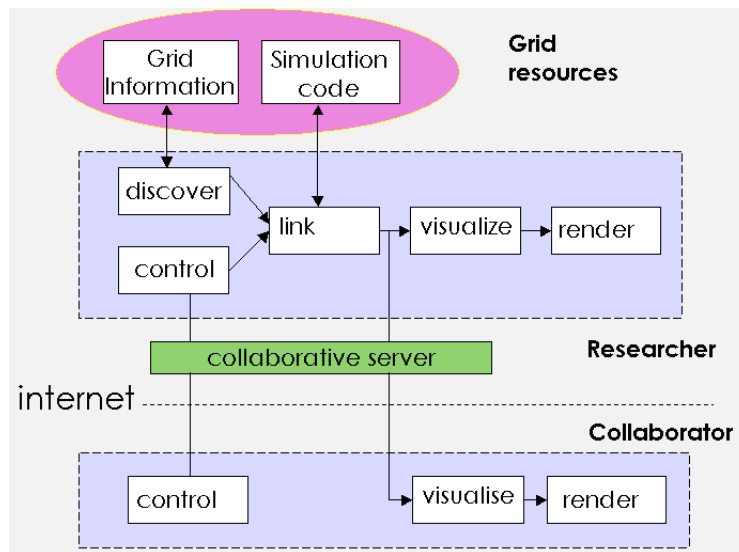


Fig. 6. Grid-based Collaborative Computational Steering

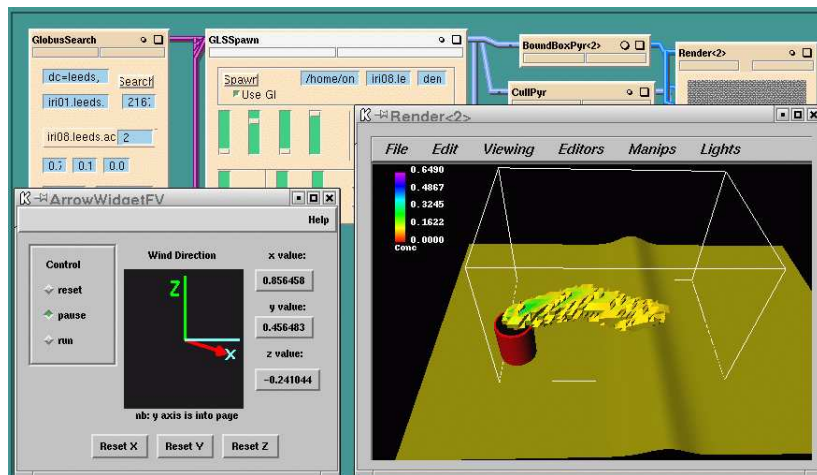


Fig. 7. Grid-based Computational Steering with IRIS Explorer

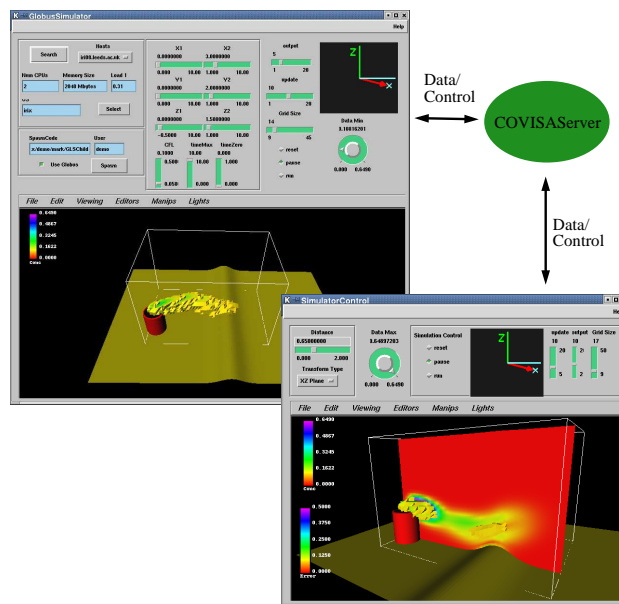


Fig. 8. Grid-based Collaborative Computational Steering Application with IRIS Explorer

This model has been realised in terms of IRIS Explorer and the COVISA collaborative toolkit. In Figure 8 we see a collaborative computational steering session in progress. The facility of IRIS Explorer to deliver end-user applications with the underlying pipeline hidden, has been used to create interfaces for the researcher and collaborator. Each user shares control of the wind velocity, the output and update time steps, the grid size for the simulation and pause/run/reset control of the simulation. In this example, simulation data is being shared between both parties with joint control over the value of the concentration level parameter for visualization. Also the collaborator is using a different visualization technique (a slice plane) to visualize an estimate of the error associated with the solution.

Note that the COVISA toolkit is cross-platform, and so the researcher and collaborator can be using different hardware and operating system.

We have run the application in a number of different scenarios

- in the simple environment of a LAN at the University of Leeds, with the simulation running on the White Rose Grid
- in a transatlantic collaboration with colleagues at CACR, Caltech in Pasadena
- in a trans-hemisphere collaboration, in which the simulation and one researcher were based at Leeds, UK in the northern hemisphere, and a collaborator was based in Christchurch, New Zealand in the southern hemisphere.

9 Conclusions and Future Work

This work has shown that the dataflow visualization environment can provide an important tool for Grid computing. Its original open and flexible design have allowed it to grow with computing technology over the last fifteen years, and it sits quite happily with the new Grid software infrastructure.

The work was implemented in IRIS Explorer, but it should be straightforward to take the same ideas and implement for AVS, or IBM Open Visualization Data Explorer.

In our experimental Grid, we take advantage of the fact that our simulations start immediately on submission. In reality a variety of job managers and queuing systems would be found in use on the Grid and work needs to be done to manage access to simulations that are scheduled for execution at a later time.

We continue to explore further ways of integrating dataflow visualization systems with Grid computing environments. In particular we are studying how pipelines can be distributed across several host machines, each running an instance of the visualization system. This was an integral part of the early design of these systems, but the original authentication mechanism has been

deprecated as being insecure and so this feature is generally no longer available. We are investigating whether this can be revived in a secure manner using Globus middleware.

There are two difficulties with our current approach:

- the simulation has to regularly poll the visualization system on the desktop to check whether any parameters have changed - this is a major overhead and slows down the simulation
- the simulation only runs as long as the visualization system is kept running on the desktop - for a long running simulation one wants to be able to have the simulation running with a life of its own, and allow the visualization system to 'check in' periodically to monitor progress

We are aiming to address these limitations in a further development cycle.

An important issue yet to be addressed in this work is data management for asynchronous collaboration. It is envisaged that users may wish to review a time-history of the computed solution together with the related steering options that were chosen at that time. The current user can then restart the simulation at a particular time and study the effects of varying some parameters of the model in detail from that point. Some of the data management issues are addressed by the Hyperscribe system already incorporated into IRIS Explorer.

The case study considered here was based around a prototype application but the developed environment is intended for generic scientific computing models. A companion demonstrator has been built, extending its use to the far more challenging simulation of elastohydrodynamic lubrication [10].

Grid-enabled visualization is an extremely important research area. Our aim in this paper has been to question the need to start afresh and develop new visualization systems for Grid computing *ab initio* - existing dataflow systems have sufficient flexibility and extensibility for them to evolve into this exciting new world.

References

1. AVS/Express website, 2004. <http://www.avs.com>.
2. E. W. Bethel and J. Shalf. Grid-distributed visualizations using connectionless protocols. *IEEE Computer Graphics and Applications*, 23(2):51-59, March/April 2003.
3. cAVS website, 2004. <http://cavs.sdsc.edu>.
4. J. Chin, J. Harting, S. Jha, P.V. Coveney, A.R. Porter, and S. Pickles. Steering in computational science: mesoscale modelling and simulation. *Contemporary Physics*, 44:417-434, 2003.
5. D.A. Duce, J.R. Gallop, I.J. Johnson, K. Robinson, C.D. Seelig, and C.S. Cooper. Distributed Cooperative Visualization - Experiences and Issues from MANICORAL Project. In *EG Workshop on Visualization in Scientific Computing*. Eurographics Association, 1998.

6. E. Engquist. Steering and visualization of electromagnetic simulations using Globus. In *Simulation and Visualization on the Grid*, pages 82–97. Springer Series LNCSE, 2000.
7. I. Foster, J. Insley, G. von Lasewski, C. Kesselman, and M. Thiebaux. Distance visualization: Data exploration on the Grid. *IEEE Computer*, December:36 – 43, 1999.
8. I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the Grid: Enabling scalable virtual organisations. *International Journal of High Performance Computing Applications*, 15(3):200 – 222, 2001.
9. Globus website, 2004. <http://www.globus.org>.
10. C. Goodyer, J. Wood, and M. Berzins. A parallel grid based PSE for EHL problems. In J. Fagerholm, J. Haataja, J. Jarvinen, M. Lyly, M. Raback, and V. Savolainen, editors, *Applied Parallel Computing Advanced Scientific Computing 6th International Conference, PARA 2002*, pages 521–530. Springer-Verlag, 2002.
11. R. B. Haber and D. A. McNabb. Visualization Idioms: A Conceptual Model for Scientific Visualization Systems. In B. Shriver, G. M. Neilson, and L. J. Rosenblum, editors, *Visualization In Scientific Computing*, pages 74–93. IEEE Computer Society Press, 1990.
12. T.J. Jankun-Kelly, O. Kreylos, K. Ma, B. Hamann, K.I. Joy, J. Shalf, and E.W. Bethel. Deploying web-based visual exploration tools on the grid. *IEEE Computer Graphics and Applications*, 23(2):40–50, March/April 2003.
13. C.R. Johnson, S. Parker, D. Weinstein, and S. Heffernan. Component-based problem solving environments for large-scale scientific computing. *Journal on Concurrency and Computation: Practice and Experience*, (14):1337–1349, 2002.
14. R. Marshall, J. Kempf, S. Dyer, and C. Yen. Visualization methods and computational steering for a 3d turbulence model for Lake Erie. *ACM SIGGRAPH Computer Graphics*, 24(2), 1990.
15. B. H. McCormick, T. A. DeFanti, and M. D. Brown. Visualization in scientific computing. *Computer Graphics*, 21(6), 1987.
16. National Grid Service website, 2004. <http://www.ngs.ac.uk>.
17. A. Norton and A. Rockwood. Enabling view-dependent progressive volume visualization on the grid. *IEEE Computer Graphics and Applications*, 23(2):22–31, March/April 2003.
18. 2004. <http://www.opendx.org>.
19. S. G. Parker and C. R. Johnson. SCIRun: A scientific programming environment for computational steering. In H. W. Meuer, editor, *Proceedings of Supercomputer '95*, New York, 1995. Springer-Verlag.
20. W. J. Schroeder, K. M. Martin, and W. E. Lorensen. *The Visualization Toolkit: An Object Oriented Approach to 3D Graphics*. Kitware, Inc., 3rd edition, 2003.
21. J. Shalf and E. W. Bethel. The Grid and future visualization system architectures. *IEEE Computer Graphics and Applications*, 23(2):6–9, March/April 2003.
22. TeraGrid website, 2004. <http://www.teragrid.org>.
23. A Tomlin, M. Berzins, J. Ware, and M.J. Pilling. On the use of adaptive gridding methods for modelling chemical transport from multi-scale sources. *Atmospheric Environment*, 31(18):2945–2959, 1997.
24. Edward R. Tufte. *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphics Press, 1997.

25. VTK website, 2004. <http://public.kitware.com/VTK/>.
26. J. P. R. B. Walton. NAG's IRIS Explorer. In C. R. Johnson and C. D. Hansen, editors, *Visualization Handbook*. Academic Press, 2003, in press. Available from http://www.nag.co.uk/doc/TechRep/Pdf/tr2_03.pdf.
27. White Rose Grid website, 2004. <http://www.wrg.org.uk>.
28. J. D. Wood, H. Wright, and K. W. Brodlie. Collaborative visualization. In R. Yagel and H. Hagen, editors, *Proceedings of IEEE Visualization '97*, pages 253–259, 1997.