Visualisation in e-Demand: A Grid Service Architecture for Stereoscopic Visualisation

Stuart M. Charters, Nicolas S. Holliman and Malcolm Munro
Visualisation Research Group
Department of Computer Science
University of Durham,
South Road,
Durham,
DH1 3LE, UK
S.M.Charters@durham.ac.uk N.S.Holliman@durham.ac.uk Malcolm.Munro@durham.ac.uk

Abstract

The visualisation strand of the e-Demand project focuses on the use of Grid services to support stereoscopic visualisation in a distributed environment. The architectural design of the framework allows each part of the visualisation pipeline, implemented as a Grid service and communicating in a peer-to-peer manner, to be highly e xible and con g-urable using the resources most appropriate to the task. An implementation of the framework has been done using Java and a case study has been developed to test it.

1 Introduction

Visualisation is an important part of many e-Science and Grid projects, it is the use of graphical representations to assist in the analysis of data. The use of stereo images in visualisation has been given a new impetus following the development of desktop autostereoscopic displays. These displays, typically modi ed LCD panels, present a stereo image to the user. The image has binocular disparity causing objects appear to leap from the screen or fall behind it. The use of these displays bring a new series of challenges to the areas of visualisation and of rendering. This paper outlines the background to stereoscopic visualisation and the technologies involved in Grid computing before discussing the proposed visualisation framework, the current implementation of it and a case study highlighting the ways in which it can be utilised.

2 Background

Many areas of computing, stereoscopic image generation, peer-to-peer computing and existing visualisation frameworks are brought together in the development of the Grid service framework for stereoscopic visualisation. These areas are examined in detail below.

2.1 Stereoscopic Image Generation

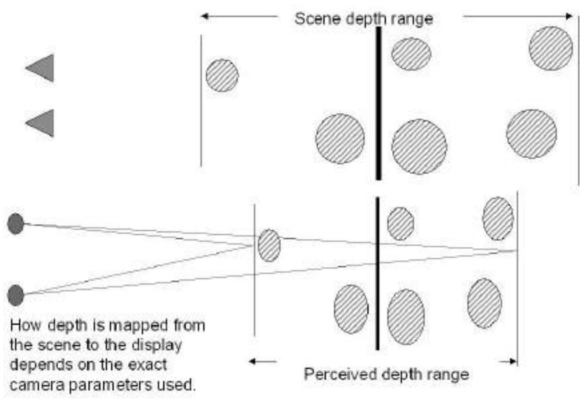
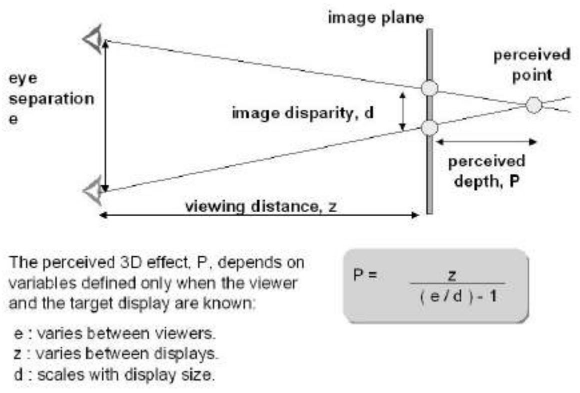

Scene depth range
How depth is mapped from the scene to the display depends on the exact camera parameters used.
Perceived depth range

Figure 1. Mapping Scene Depth to Perceived Depth

Stereoscopic 3D displays are becoming increasingly widely available which is in large part due to recent advances in auto-stereoscopic 3D displays [8]. An auto-stereoscopic 3D display allows a viewer to see a stereoscopic 3D effect without wearing any special glasses, such



The perceived 3D effect, P, depends on variables defined only when the viewer and the target display are known:

$$P = \frac{z}{(e/d) - 1}$$

e : varies between viewers.
z : varies between displays.
d : scales with display size.

Figure 2. The Perceived Depth Equation

as polarising or LCD shutter glasses. As well as being more acceptable to users these new technologies can be integrated in desktop, laptop and mobile devices at relatively low cost.

While there has been rapid advances in 3D display technology, the image generation methods needed to use the displays require further work. Stereoscopic images are often generated on an ad-hoc basis using rules of thumb that have limited basis in theory, for example simply setting stereo camera separation to be eye separation is almost always incorrect. Such approaches provide limited control over the resulting 3D effect and this in turn often results in images with too much or too little 3D effect.

Recent research [15], [10] has addressed these issues and new algorithms for stereoscopic image generation are beginning to account for the user's perception of depth in the nal images. The new algorithms consider the stereoscopic image generation process as a mapping of depth from a 3D scene to a perceived depth range on a target display [10]. They then ensure the desired mapping is achieved by calculating exact stereoscopic camera parameters.

In order to provide better control over the 3D effect the new algorithms need to know more about the target 3D display and the ultimate viewer. This is because, as shown in gure 2, the perceived depth depends on parameters only de ned when the viewer sits down in front of a speci c display and looks at the image. A consequence of this is that 3D images are not portable between displays and should ideally be generated when the viewing situation is known.

The requirement for just-in-time image generation to control the 3D effect makes Grid services particularly well suited to stereoscopic visualisation tasks. The compute resources to generate the correct images for the current viewing situation can be assembled on demand and the appropriate image delivered to the viewer.

2.2 Peer-to-Peer Computing

Peer-to-peer computing has been established as a good mechanism for data sharing with the development of such systems as Gnutella [2] for sharing data les distributed around the internet. Different peer-to-peer systems offer different bene ts, systems like Gnutella excel in easy data sharing and fast access to data, other systems such as Chord [14] guarantee that les can be located whilst accepting an overhead when adding and removing les. Other peer-to-peer systems exist for the sharing of CPU cycles such as SETI@Home [3] and Entropia [1].

In the peer-to-peer model each node in the network acts as both a server and as a client. This allows communication between nodes to be conducted directly rather than via a third party, as in the traditional client/server model where the server controls everything. The peer-to-peer model allows the resources of each host to be utilised to the greatest extent possible if each node provides metadata about its capabilities. This model is already well established in distributed computing systems [4] and graphics applications on parallel computers [9].

Peer-to-peer computing has been growing alongside the Grid and only recently have the principals of peer-to-peer computing been applied to Grid software. The next developments in Grid software will move further towards the peer-to-peer model of computing. Grid and peer-to-peer computing are beginning to converge [6] and this convergence will bring bene ts to both communities.

2.3 Existing Visualisation Systems

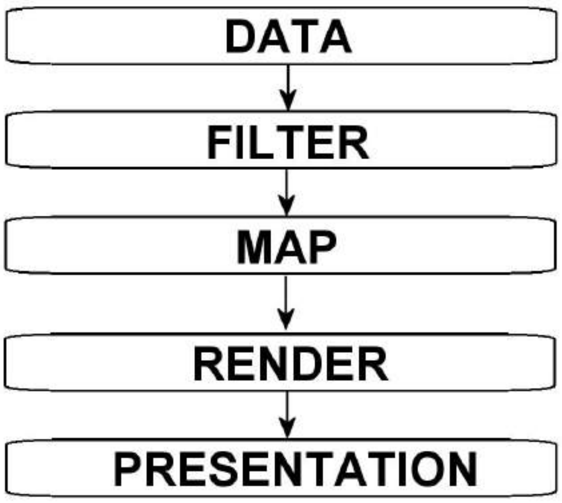


Figure 3. Traditional Visualisation Pipeline

Current visualisation systems, are built using a range of different architecture models ranging from early monolithic architectures through Modular Visualisation Environments to Client/Server and Web based systems. These systems all follow the traditional visualisation architecture as described in gure 3, dividing up the total visualisation process into different subprocesses.

Modular visualisation environments (MVEs) such as IRIS Explorer [13] and AVS/Express [5] demonstrate the subdivisions in the visualisation pipeline through the use of modules that are linked together in an integrated environment. This environment is typically run on a machine with suf cient memory, processor and graphics capability to handle each stage of the pipeline, this can result in the machine being under utilised for much of the time that the visualisation is being run.

3 Proposed Architecture

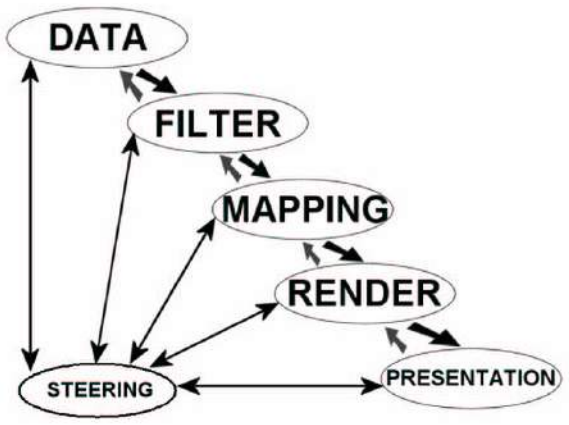


Figure 4. Proposed Architecture

The proposed architecture outlined in gure 4 for Grid visualisation builds upon the traditional visualisation pipeline, preserving the traditional forward o w of data between parts of the pipeline but adding a reverse o w of information between services, allowing for con guration of, and, control to be exercised over, the whole pipeline.

The design allows for peer-to-peer communication between Grid services that can be distributed across many different locations, this differs from other Grid visualisation initiatives which operate in a client/server manner. The peer-to-peer communication model theoretically reduces the communication overhead by 50% compared to the client/server model as traf c is transferred between services and not via a server, which would result in information being transmitted twice over. The service approach to visualisation allows each part of the visualisation to be run on a different machine or machines with appropriate computation power, storage or graphics capability. This allows for greater utilisation of resources on the Grid helping to reduce the total cost of ownership and increase the e xibility of the visualisations.

Modular visualisation environments are fully self contained with the visualisation design tool, visualisation execution environment and the end user display all contained within a single software environment.

3.1 Metadata, Grid Service Data

The visualisation framework has a provision within each service, through the use of Grid Service Data items to provide metadata to the user and to other Grid services about the service. This provision allows services to con gure themselves in response to the other services they are interacting with. It allows for some type checking to be performed so that different datatypes are not mixed and services incorrectly composed. This service data could also provide more detailed information about the capability of the service and therefore be used in contract negotiation between services where quality of service, security, trust and privacy are important.

One use of the metadata is the con guration of a render service within the visualisation pipeline to support a presentation service running on an autostereoscopic display. Parameters required to generate the correct stereo pairs for the display will be passed to the render services.

3.2 Visualisation Composition

The framework allows for the decoupling of visualisation design or composition from the end user this has the bene t of allowing a visualisation expert to construct a visualisation for end users where ever they may be in the world.

By creating the visualisation using Grid services the visualisation can persist on the Grid, allowing for different time, different place collaboration of experts in the construction of the visualisation.

The remote composition and con guration of the visualisation allows the end user to make more effective use of the visualisation and of their time, allowing them to direct their energies to the analysis tasks that are being performed.

Once the visualisation has been composed it exists on the Grid and the expert can move to other tasks without the need to have running a copy of the composition tool.

3.3 Steering / End User Control

It is recognised that end users of visualisations have a need to exercise control over the visualisation pipeline, to this end the provision for steering of each stage of the pipeline is provided. It is envisaged that this interface be separate from the presentation service and that it does not need to be in existence for the duration of the visualisation and can be invoked as-needed.

The mechanism for control over the visualisation is consistent across all tools that are used to manipulate the visualisation and the pipeline this is through the implementation of a steering interface by each service in addition to the stage interface provided.

3.4 Advantages

There are several advantages to the design proposed, these are discussed below.

3.4.1 Scalability at each stage of pipeline

The service based approach allows each part of the pipeline to be executed on different logical machines in different geographic locations. This separation also allows for each service to be scaled in the most appropriate manner for the computation that it is to perform, for example a render service can be hosted on a high performance graphics machine to produce high quality images. Whilst the mapping service could be executing on a parallel computing array accessing data and lter services running at a high capacity storage farm. The option to make use of scavenger grids, where spare cycles are used for computation is also available.

3.4.2 Separation of visualisation designer and visualisation user

The ability for one user to build or compose the visualisation at one time in one place and then for another user to make use of the visualisation at a different time in a different place has enormous bene ts especially as collaboration between scientists based in different geographical locations becomes increasingly common.

3.4.3 Enhanced Collaboration

The e xibility of the design, allowing branching at each stage of the pipeline, makes collaboration between parties, both same time different location, and different time different location, easy to establish. Different locations can run different rendering services to support the display devices they have, alternatively a single render service can support multiple presentation services.

Collaboration is an important use of visualisation tools, allowing images to aid communication between scientists allowing different experts to come together and examine data that requires more than just their expertise.

3.4.4 Generate the correct stereo images for the display determined at run-time

The proposed architecture has bene ts for stereo visualisation as it allows the rendering and presentation services to be con gured to generate stereo pairs that are correct for the display that is being used. The ability to describe services using meta-data which can be sent in the reverse direction up the pipeline, for example from presentation service to render service, will enable support for high quality stereoscopic image generation. Using the meta-data describing the target display and viewing situation the renderer will now be able to generate images that have equivalent 3D effect on different displays and for different viewers. It is important to manage the stereoscopic 3D effect in this way in all types of visualisation where users are using 3D shape judgements to understand their data.

4 Success Criteria

Liefer and Pierson [11] present a list of six requirements for a distributed visualisation service, these requirements provides an initial set of functionality against which to evaluate the proposed distributed visualisation architecture

- Web based
- Transparency
- Customizable Information Delivery
- Open Software
- Distributed/Roaming
- Quality of Service

4.1 Web Based

This is expanded by Liefer and Pierson to mean reliance on well-established standards for communication between user and service. By exploiting the Grid and the work being done to create standards for Grid services our architecture ensures that it will be usable and accessible.

4.2 Transparency

”making the visualization technology at the disposal of every Internet surfer implies to make as less changes as possible to his habits”. [11]

The architecture by default does not ensure this requirement or but does promote it, as the Grid becomes more wide spread and the use of visualization becomes more of an everyday occurrence research in this area should insure that this goal is achieved. The architecture lends itself to automatic discovery and composition by agents to present a solution to the user but this must be reserved as a future goal.

4.3 Customisable information delivery

This can be summarised as providing the best solution with whatever resources are available. The architecture supports this to a high degree, the ability to replace any part of the pipeline means the most suitable part for the job can be used which should result in the best possible solution for the available resources. The use of meta data will ensure the correct stereoscopic images are displayed to the user.

4.4 Open software

Integration of new components without re-engineering the whole system. The Grid environment is ideal for this type of addition, new services can be added at any time without any impact on users. The only time that users may be impacted is if the protocol and standards for communication between the rendering service and presentation client changed. This would require an upgrade to the presentation client, an upgrade would also be required if the discovery mechanisms changed, however this would have a very wide reaching impact across all Grid services.

4.5 Distributed / Roaming

This requirement can be split into two parts:

a) the right resources for the service

b) the right resources for the service and the right data to analyse

Case a is solved by the Grid subsystems, either a service runs on a x ed resource or the Grid scheduling system will pro le a service and locate the 'best' resource to run it on.

Case b is more complex, in part it is solved by use of discovery mechanisms and the solution to case a above however a complete solution is more likely to come about with an implementation that supports automated lookups.

4.6 Quality of Service

This requirement relates to security, trust and quality of service/data. Security and trust are both issues that are being addressed by the Grid community and the results of their research should provide a solution in this area. The quality of a service or of a dataset is in part addressed by the security and trust issues but also by the information a service provides about itself, it is therefore the users job to evaluate the acceptability of the level of quality the service provides.

4.7 Summary

Overall the majority of each requirement is satis ed by either the Grid environment or the architecture design itself. Those parts that are not covered are the focus of ongoing research and are issues that should not stop the implementation and adoption of the architecture in the short term as they do not impact on the functional aims of the architecture.

5 Current Implementation

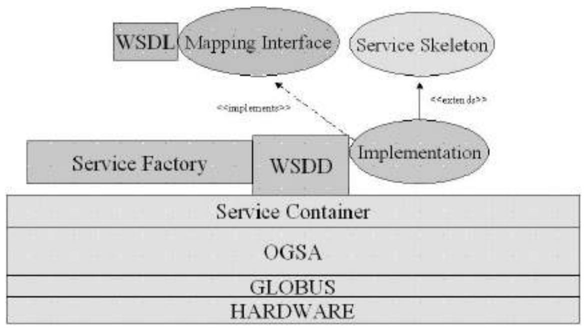


Figure 5. Service within an OGSA Environment

The current implementation of the visualisation framework concentrates on turning the traditional visualisation pipeline into a distributed service framework. Figure 5 shows an example of a service and how it ts within the OGSA (Open Grid Services Architecture) framework [7]. The service interface and WSDL (Web Services Description Language) document are provided as part of the framework with the service implementation and service factory being the responsibility of the developer. The service skeleton is provided as part of the OGSA framework as is the service container that all services run in.

Implementation to date has been done using Java as the language of choice with the Alpha-3 implementation of OGSA Toolkit. The WSDL provides a platform and language independent XML based description of the interface to each service. A concrete implementation of each WSDL interface is provided through a Java interface. The interfaces provide two methods, one to set the data source from which the service should pull data and a second method which allows another service to get the data from that service. The Service Skeleton is an abstract Grid service which contains much of the boilerplate implementation allowing the service implementation to concentrate on the task the service is performing. For each service a service factory is also constructed, this factory is responsible for 'manuf acturing' instances of a service on a host. The WSDD (Web Services Deployment Descriptor) ties the Service Factory, the WSDL and the service to the service container. The service container acts as a sandbox execution environment for services. It is responsible for the management of other software and physical resources and the host security policy. Communication between services is performed using the Simple Object Access Protocol (SOAP) which operates over connectionless protocols such as HTTP.

To demonstrate and test the framework as implementation has progressed a case study has been used.

5.1 Chemistry Case Study

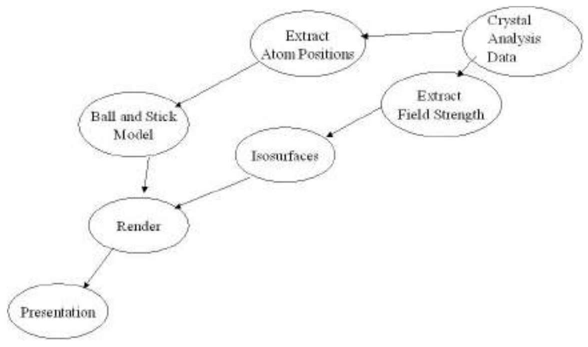


Figure 6. Design for X-Ray Diffraction Crystallography

A case study has been taken from the chemistry domain, speci cally X-Ray Diffraction Crystallography, and has been used to demonstrate and test the framework. The application requires the transformation of a small data le into a ball and stick representation of a molecule and an isosurfacing of electric eld strength within the molecule. Figure 6 shows the design for the chemistry case study, it is currently partially implemented to create the ball and stick model of the molecule.

The data le is loaded by the data service and then transferred to the lter service. The lter service extracts the data relevant to the ball and stick model which is loaded by the mapping service whose responsibility it is to create a geometric representation of the model. The model is then transferred to the render service which sends the nal image through to the presentation service where it is displayed to the user. The gure demonstrates the practical application of the visualisation framework, it highlights several important features for example, parallel branches that are recombined to form the nal output at the render service before transmission onto the presentation service.

5.2 Initial Results

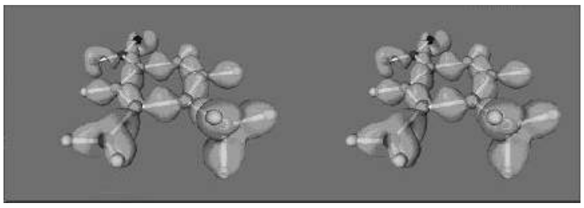


Figure 7. Initial Fast Track Results

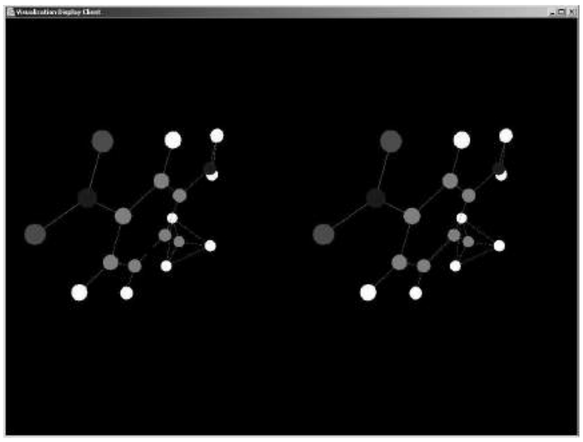


Figure 8. Initial Deep Track Results

The chemistry case study took a two track approach to the project, a fast track and a deep track. The fast track approach provided a method to generate example images to show the results that the nal project aims to produce. This was done using currently available visualisation packages (IRIS Explorer) and a graphics package. The deep track approach allows the computer science aspects of the project to be investigated more fully whilst having a reference to evaluate the generated output against and a clear idea of the requirements of the client. Figure 7 shows initial fast track results generated using IRIS Explorer and using a graphics package. This fast track result is an example of the type of output the framework will product, a current deep track output is shown in Figure 8 for comparison.

6 Future Work

The current implementation of the framework has demonstrated the basic principals behind the architectural design. Research and development work continues to implement features to ensure the framework is robust, reliable, e xible and deployable.

6.1 Communication

To achieve the desired levels of flexibility and configuration bi-directional communication throughout the pipeline will be implemented. To allow this flexibility to be exploited, further work on the peer-to-peer communication between services needs to be done as part of the development. Examination of alternative protocols to SOAP such as JXTA needs to be performed and a suitable communication mechanism established.

6.2 Lookup and Discovery

Lookup and Discovery using UDDI is an area requiring attention to support the remote composition of visualisations. When using the current implementation of the framework a visualisation user must know the location of each service in the pipeline when the framework supports lookup and discovery the user will be able to search for services that match their requirements.

6.3 Composition

Following implementation of Lookup and Discovery an environment to support composition of visualisations needs to be developed. This environment may resemble those of current MVEs and will support the composition of complete visualisation pipelines.

A further development of the composition process would be a move towards automatic or semi-automatic data discovery and visualisation composition, supported by technology such as Agents.

6.4 End User Interface/Steering

With the implementation of bi-directional communication in the visualisation framework coupled with developments in Grid standards, steering of services will be addressed.

As the visualisation designer and the visualisation end user are decoupled from the framework an interface to support the end user interacting with the visualisation framework, to allow steering and configuration to be performed will be developed.

6.5 Security

To take increase security within the framework the mechanims for security in place on the Grid will be exploited. These mechanisms will allow restricted access to services, provide secure communication to services and help build virtual groups and virtual organisations for collaboration using visualisations.

6.6 Robustness

As with many developments the initial focus for the framework has been on testing concepts and principals, evaluating ideas and experimenting. As the framework matures the focus will change from feature implementation to ensuring the robustness of the framework, other strands in the e-Demand project, for example, OGSA-FIT [12] to allow fault injection, will help to test and evaluate the framework.

6.7 Service Development

For a visualisation solution to gain acceptance and to be of use it must be able to put into operation 'out-of-the-box' to achieve this end services at each stage of the pipeline must be developed and deployed in a manner that is easily accessible to users of the system. One method of achieving this rapidly would be to provide wrapper classes to allow modules from existing visualisation systems to be ported to the framework.

As the visual output is the part of the framework that a user will see in operation most often, the render service is one that will require development effort to ensure that high quality timely output is achieved.

6.8 Grid Standards

As Grid standards are still evolving development of the framework will be driven in part by these changes. Some parts of the OGSA specification are going through international standardisation processes, GSDL (Grid Services Description Language) is one such example. As these parts of the specification become stable they can be incorporated into the framework implementation. In the case of GSDL, this will be migrated to from WSDL and will assist in the implementation of multiple interfaces to support steering.

7 Conclusion

The paper outlines a proposed architecture for Grid based stereoscopic visualisation and the reasoning behind that architecture. Using Grid services and peer-to-peer communication a flexible and scalable framework has been developed to implement the architecture. The design allows decisions about the visualisation to be made at the point of delivery which is a requirement for the use of stereoscopic display technology where stereo pairs are not easily transferable between displays. The current implementation status of the framework is highlighted through the use of a Chemistry case study and the initial results from fast track and deep track approaches to the problem are reported.

8 Acknowledgements

This work is funded by the EPSRC/DTI funded e-Demand: A Demand-Led Service-Based Architecture for Dependable e-Science Applications Project (Grant No.: THBB/C008/00112C). The project is supported by Industry partners Sharp, Sun and Sparkle Technology.

Thanks also to the Durham Crystallography Group particularly Professor Judith Howard and Mike Probert for providing and supporting our use of the example data set. http://www.dur.ac.uk/crystallography.group/

References

[1] Entropia. *http://www.entropia.com/*, 2003.

[2] Gnutella protocol speci cations. *http://www.ovmj.org/GNUnet/papers/gnutella_protocol.pdf*, 2003.

[3] Seti@home. *http://setiathome.ssl.berkeley.edu/*, 2003.

[4] S. Ahuja, N. Carriero, and D. Gelernter. Linda and friends. *Computer, 19(8):26–34*, August 1986.

[5] AVS. Avs/express. *http://www.avs.com/*, July 2003.

[6] I. Foster and A. Iamnitchi. On death, taxes, and the convergence of peer-to-peer and grid computing. *2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)*, 2003.

[7] GGF. Open grid services architecture: A roadmap. *http://www.ggf.org/ogsa-wg/ogsa_roadmap.0.4.pdf*, 2003.

[8] N. Holliman. 3d display systems. *Handbook of Optoelectronics, Institute of Physics Press, ISBN 0 7503 0646 7*, November 2003.

[9] N. S. Holliman, C.-M. Wang, and P. M. Dew. Mistral-3: parallel solid modelling. *The Visual Computer, pages 356-370*, July 1993.

[10] G. R. Jones, D. Lee, N. S. Holliman, and D. Ezra. Controlling perceived depth in stereoscopic images. *Stereoscopic Displays and Virtual Reality Systems VIII, San Jose, California, SPIE Vol. 4297A*, January 2001.

[11] W. Lefer and J.-M. Pierson. Visualization services on the web. *SNPD'00 (International Conference on Software Engineering Applied to Networking and Parallel/Distributed Computing)*, May 2000.

[12] N. Looker and J. Xu. Assessing the dependability of ogsa middleware by fault injection. *22nd Symposium on Reliable Distributed Systems,Florence, Italy*, 2003.

[13] NAG. Iris explorer. *http://www.nag.co.uk/Welcome_IEC.html*, July 2003.

[14] I. Stoicaa, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *SIGCOMM2001, San Diego*, 2001.

[15] Z. Wartell, L. Hodges, and W. Ribarsky. Balancing fusion, image depth and distortion in stereoscopic head tracked displays. *Computer Graphics, Proc. ACM Siggraph99*, 1999.