A JXTA-based architecture for 3D distributed visualization: D3D

C. Zunino[†] **A. Sanna**[†]

Dipartimento di Automatica e Informatica,
Politecnico di Torino, corso Duca degli Abruzzi 24,
I-10129 Torino (Italy)

[†]**{c.zunino, sanna}@polito.it**

ABSTRACT

Entertainment, e-learning, scientific visualization, CAD, are only some disciplines where the capability of visualizing 3D complex objects is important. On the other hand, several new technologies allow users to share files (but more in general resources) using peer-to-peer (P2P) paradigms. In this paper we define a peer-to-peer software framework which supports the design of three–dimensional multi–user environments. In particular a P2P technology is used to design and implement a distributed visualization system where many users can concurrently inspect and analyze a virtual scene. The Sun Microsystem JXTA framework is used to develop the distributed network infrastructure while the graphics interface is implemented by the Java 3D libraries.

Keywords: Distributed visualization, peer-to-peer, 3D objects sharing.

1. INTRODUCTION

In the last years Internet has been used for different purposes: entertainment, e-learning, e-commerce, collaborative work, and many others. Entertainment is a sector where 3D representations are extremely important: for example there are online communities like ActiveWorlds, or 3D games where each user challenges with others players on the network.
Also the e-learning is becoming a practical way for education. In this context introducing the 3D can improve clearness and making available the interaction with the studied model, for instance a car engine or another complex object.
Moreover in these years another type of technology is growing up: the peer-to-peer (P2P).
P2P allows to design and implement large distributed systems devoted to share resources (mainly files) among peers. The term peer denotes a node of the network, in general, able both to ask the other nodes for services (the peer acts as a client) and to provide itself a service (the peer acts as a server); some P2Ps denote peers using the term *servent* (Gnutella).

This work aims to use a P2P approach in order to design and implement an efficient and highly scalable multi-user distributed 3D visualization platform. The main goal is to realize a system to remotely share 3D models allowing several users to interact with them. The system supply also a virtual environment where users can meet and communicate. Each user is represented in the virtual world by an avatar that can move within the scene. A user can load into the world objects described in 3DS and VRML formats; an object can be done visible to the other users sharing the same visualization session. The application is developed using Java technologies, in particular the JXTA architecture to implement the P2P network and the Java3D libraries to render and manipulate objects. The advantage of using Java3D is due to the support of the hardware acceleration of the latest graphics adapters.

A set of goals have been considered in the design phase:

- Scalability: the adding of new nodes (peers) to the architecture has to be an easy task and the system performance must not be affected in a noticeable way.
- Platform independence: the access to the system has to be possible using almost any platform (Windows, Unix/Linux, Mac, and so on).
- Modularity: the whole system is designed as a module collection and each module can be accessed exclusively by its own interface.

Section 2 presents an overview of previous work on distributed visualization while a brief review of the JXTA technology is presented in Section 3. Section 4 describes the logical network infrastructure of the proposed framework, while a detailed description of the peer architecture is provided in Section 5. Remarks on performed tests are presented in Section 6. Finally, Section **Errore. L'origine riferimento non è stata trovata.** outlines conclusions and future work.

2. BACKGROUND

The features and advantages of the distributed visualization can be summarized in three main concepts:

- Flexibility: often visualizations are made by visualization experts, distributed visualization can bring visualization to people who are experts in other areas or incidental users.
- Decentralization: last generation PCs are powerful enough to perform visualizations that only a few years ago would have required a specialized graphics workstations. Distributed visualization allows people to create visualizations on their desktop computers, from data that are retrieved from servers on the local network or on the Internet.
- Mobilization: the distributed visualization allows the user to create a visualization anywhere. All the user needs is a computer or a similar device and a network connection. The user can work at home or on the road, or he/she could present the results of a visualization to a client at the client's location.

One of the initial works on distributed architecture for visualization was presented by Ang *et al.* [2]: VIS. VIS is a visualization tool that distributes the generation of 3D models from volume data amongst available hardware.

Engel *et al.* [6] used local low-end desktop and remote high-end graphics hardware for interactive visualization of tomographic image data combining local, remote, and hybrid rendering techniques. In [7] a CORBA-based connection between a finite element solver PAMCRASH (from ESI Group, France) and the visualization tool CrashViewer is addressed.
In [15] and [18] a distributed system for visualization and sonification of scientific data based is implemented using respectively CORBA (Common Object Request Broker) and an XML-based SOAP (Simple Object Access Protocol) protocol.

A tool for geometric algorithm designers and for educators who teach algorithms to non-experts was developed by Lee, Sheu and Shen [12] extending Xfig with a message-driven interface and a socket-based interprocess communication.
Visapult is a parallel visualization tool that employs Grid-distributed components, latency tolerant visualization and graphics algorithms, along with high performance network I/O in order to achieve effective remote analysis of massive datasets. In [3] an improvement to network bandwidth utilization and responsiveness of the Visapult application is discussed.
Another system [20] presented a platform-independent visualization system for visualizing particle tracing in computational fluid dynamics, using a distributed architecture based on Jini.

In the last years an innovative network architecture has been developed: the peer-to-peer (P2P). Such architecture is characterized by a direct access between peer computers, rather than through a centralized server. In particular a P2P system is different from the traditional client/server model because the applications involved act as both clients and servers: while they are able to request information from other servers, they also have the ability to act as a server and respond to requests for information from other clients at the same time.
It is possible to classify P2P systems using two general aspects: the degree of centralization and the network structure.

A pure P2P application has no central server [5]. In a P2P partially centralized architecture there are some nodes that assume a more “important” role than the rest of the nodes, acting as local central indexes. Some architecture of this kind are Morpheus, (http://www.morpheussoftware.net/) and Kazaa (http://www.kazaa.com/).

Finally, there are P2Ps with a centralized server (e.g. Napster). In unstructured networks, the placement of data is completely unrelated to the overlay topology. On the opposite, in structured networks, (such as CAN [16], Pastry [17], Chord [19], and so on) the overlay network topology is tightly controlled and data (or pointers to them) are placed at precisely specified locations. For a deeper description on peer-to-peer topologies see [1].

3. A BRIEF INTRODUCTION TO JXTA

JXTA [4], [9] is an open-source project (http://www.jxta.org/) that defines a set of protocols for ad-hoc,
pervasive, peer-to-peer computing. The Project JXTA protocols establish a virtual network overlay on top of the Internet and non-IP networks, allowing peers to directly interact and organize independently of their network location. The Project JXTA software architecture is divided into three layers:

- The Platform Layer (JXTA Core). The platform layer, also known as the JXTA core, encapsulates minimal and essential primitives that are common to P2P networking. It includes building blocks to enable key mechanisms for P2P applications, including discovery, transport (including firewall handling), the creation of peers and peer groups, and associated security primitives.
- The Services Layer. The services layer includes network services that may not be absolutely necessary for a P2P network to operate, but are common or desirable in a P2P environment. Examples of network services include searching and indexing, directory, storage systems, file sharing, distributed file systems, resource aggregation and renting, protocol translation, authentication, and PKI (Public Key Infrastructure) services.
- The Applications Layer. The applications layer includes implementation of integrated applications, such as P2P instant messaging, document and resource sharing, entertainment content management and delivery, P2P Email systems, distributed auction systems, and many others.

The JXTA network consists of a series of interconnected nodes, or peers. Peers can self-organize into peer groups, which provide a common set of services. In particular, JXTA denotes two particular peers called: rendezvous and edge. Rendezvous peers can be used to interconnect different groups logically separated. A rendezvous peer acts as a router; all (edge) peers connected to a rendezvous can interact with peers not belonging to the group only sending and receiving messages via a rendezvous peer. Rendezvous peers naturally allow to build a controlled topology (when a peer is set as rendezvous a list of other rendezvous peers to be used for setting up direct connections has to be specified) able to manage all communications needed for the proposed system.
Moreover, a sort of fault tolerance capability is already provided in JXTA; for instance, if a rendezvous crashes all edge peers connected to it can be automatically “redirected” to another rendezvous preserving system functionality. In JXTA is also implemented a loosely-consistent DHT walker approach for searching advertisements and routing queries in the JXTA rendezvous network. The loosely-consistent DHT walker uses an hybrid approach that combines the use of a DHT to index and locate contents, with a limited range walker to resolve inconsistency of the DHT within the dynamic rendezvous network. This approach has the advantages of not requiring a strong-consistency DHT maintenance, and is well adapted to ad-hoc unstructured P2P networks.

More details concerning JXTA can be found in [4], [9].

4. NETWORK ARCHITECTURE

A general description of a distributed visualization system can be: *using one or more computers to give one or more users an image of a data set*. There are many ways to do this, as described in [10]:

- Parallel visualization: the time needed for a visualization can be reduced using parallel CPU architectures.
- Multiple source visualization: results from multiple data sources are combined.

- Network-oriented visualization: enables users to create visualizations from data on a remote server.

There is also another kind of distributed visualization, the collaborative visualization: it is useful for a team, where every member is looking at the same subject, which is often steered by one team member.
The proposed framework is based on this concept: the main objective is to create a virtual environment where users can meet, communicate and share 3D objects. Each user can move everywhere (a collision detection mechanism avoids user-object and user-user intersections), can communicate using a chat and can inspect 3D objects using keyboard and mouse. Performance of three-dimensional visualization environments depend on network connections.
Funkhouser [8] and Macedonia . [14] describe the classification of network connections and network topology. Mainly network connections can be divided into two groups: there are server based connections, for example [11], and peer-to-peer based connections [13]. In a peer-to-peer based environment, each client has to maintain connections to all other clients. Multicast can be used to pass information to a group of clients. If the network is not multicast capable, then single connections have to be used: if the environment should support a lot of clients an according amount of connections is needed.

The proposed framework goes beyond this limit because uses multicast for local area network and single link between far peers.
The D3D (Distributed 3D) architecture uses the two main kinds of JXTA peers: rendezvous and edge (see Section 3).

In particular, a typical network configuration has to be composed by one rendezvous peer for each local area network (LAN) and possibly many local edge peers. JXTA uses a multicast protocol to discover other peers in a LAN, while in a geographically distributed network rendezvous peers act a sort of routing mechanism allowing nodes placed in different LANs to communicate.

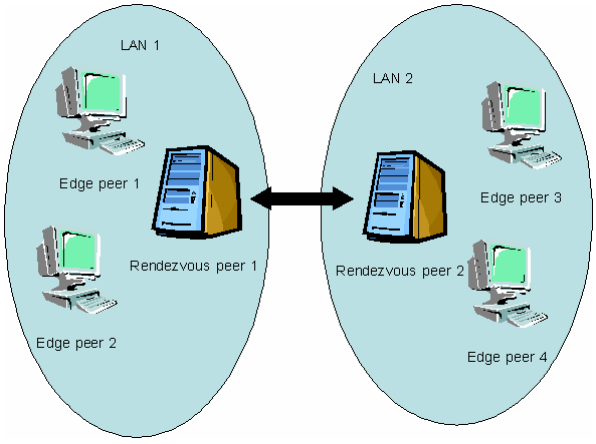

Figure 1: A typical D3D network

Edge peers can interact with peers not belonging to the group (a group is leaded by at least a rendezvous peer) only sending and receiving messages via a rendezvous peer. As shown in Figure 1, where a typical D3D scenario is depicted, the edge peers in LAN 1 can communicate with each other and with Rendezvous 1, while to communicate with Edge peer 3 and Edge peer 4, they have to use Rendezvous 1 that, communicating with Rendezvous 2, forwards messages to the destination peers.

Rendezvous and edge peers differ only for a configuration of the underlying JXTA layer, but from the user point of view, they present the same interface. In the simplest configuration, the rendezvous peer is the node starting the distributed visualization session, while other users "enter" in this section by connecting to the system as edge peers.

5. THE PEER ARCHITECTURE

The system is developed to obtain scalability and easiness to develop. For this purpose the application is subdivided into packages that can interact trough interfaces. In particular, the main packages, as shown in Figure 2, are five:

1. GUI (Graphical User Interface): this package contains all the classes to manage the graphical user interface.
2. J3D: this package contains all the classes to manage the virtual environment.
3. JXTA: this package contains all the classes necessary to communicate with remote peers. It contains also the XMS service described below.
4. Background: this package contains all the classes to manage the application back-office. For instance, it contains the watcher of used memory.
5. Util: this package contains all the classes useful for all packages interaction.

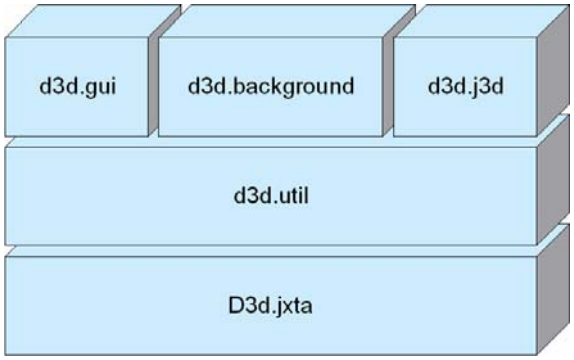

Figure 2: The peer architecture

This packages subdivision allows also to support different data format. The output of some applications can be only standard graphic objects such as polygon meshes while other applications might use more sophisticated objects (like 3DS or VRML). The J3D package can visualize different file formats and, if it will be necessary to support other formats, new software modules could be added to this package without any change in other parts of the architecture.
JXTA supplies the CMS (Content Management System) to manage shared resources in a distributed environment. Starting from the CMS, another component has been created for the proposed system: the XMS (eXchange Management System). This component, unlike the CMS that permits only file sharing, supplies the exchange of messages and 3D objects. In particular XMS allows resources sharing, 3D objects download from remote peers and peer searching (to find all peers participant to a visualization session).
Because there is no a central list of all available resources each peer, using XMS, has to maintain and exchange the list of its resources. Moreover, when a peer wants to share a 3D object,

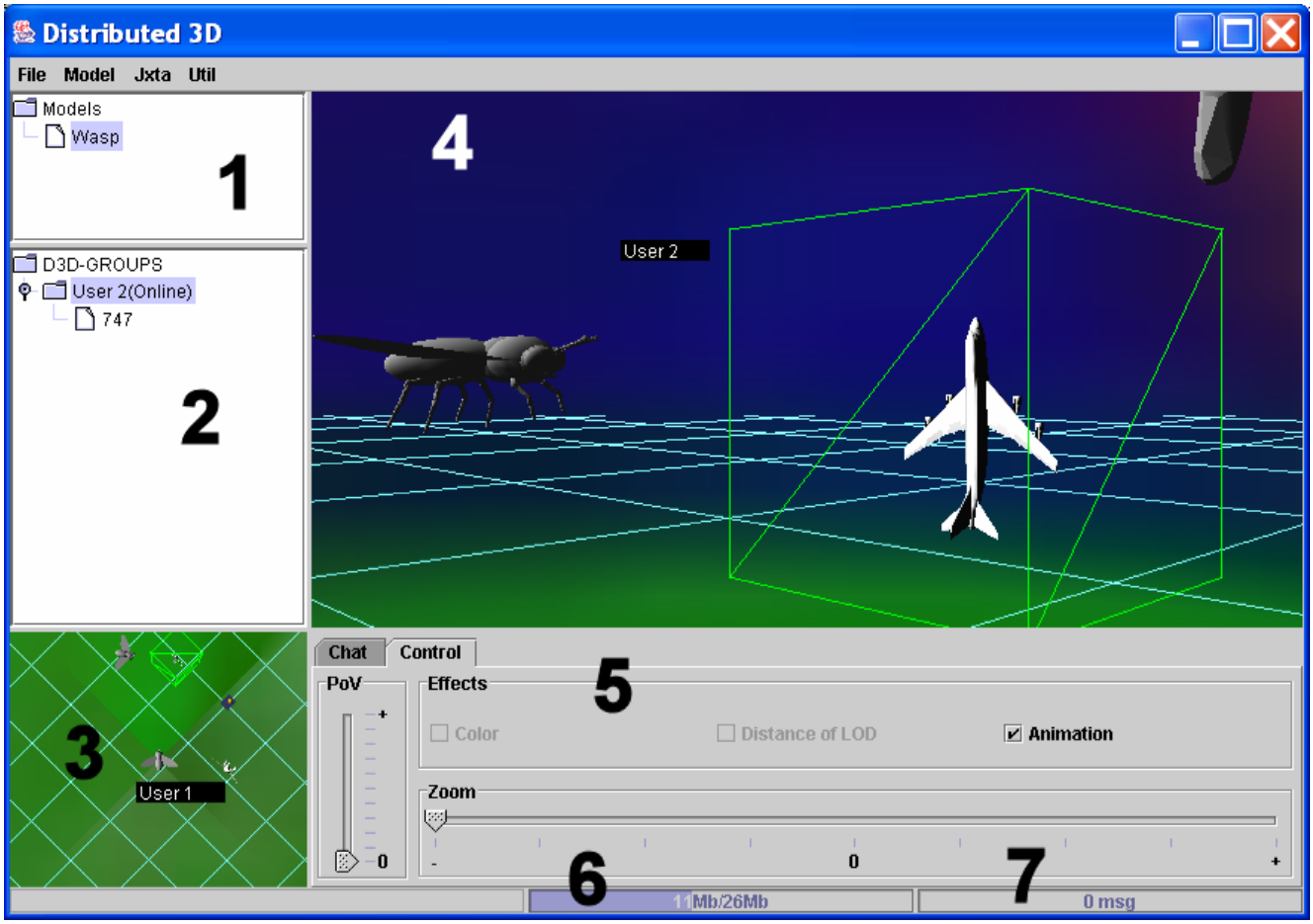

Figure 3: The D3D graphics interface

XMS supplies a pipe mechanism to download the model. Finally, XMS, during the starting phase, is responsible to send all information to all other peers in the network.

Each peer sends asynchronous messages to all other peers to communicate every action in the virtual environment (an avatar movement or a roto-translation of an object loaded in the scene). This can cause an overload due to manage too many messages. To solve this problem a component has been developed; in particular, a manager based on a FIFO queue is used. This manager allows to obtain two advantages:

- The message load generated by each peer is independent of the number of received messages. In this way a peer can carry out other operations, while the manager can process messages.
- Messages loss (and hence the coherence in the virtual world) is avoided.

This mechanism causes a delay tied to the network overload, but this drawback is overcome by the reliability it inserts. Finally, to have both a useful and nice to see Graphical User Interface, the system offers seven main areas to interact with other peers and to move in the virtual environment.
As shown in Figure 3, the window can be divided in seven areas:

1. In this area there is a list of the 3D models loaded. The user can manage and obtain information using a popup menu. In the example of Figure 3 users are represented by wasp models.
2. In this area all the connected peers are shown, as well as the shared models. In Figure 3 “User 2” shares a model of an 747 airplane. The user currently managing an object sees the object itself encapsulated into a green bounding box; on the other hand, the other users see the same model encapsulated into a red box. A simple click with the mouse on the model allows to release the control. In Figure 2 the “User 1” controls the model.
3. In this area a map of the virtual environment is presented. This is very useful for large virtual worlds to easily find objects and users placement.
4. This area shows the peer/user point of view. The user can interact using keyboard to move the avatar and mouse to manage 3D objects. A collision detection mechanism avoids a user could penetrate objects or other users.
5. In this area there are two tabbed panes: the former is for the chat system, while the latter is to control some parameters: the vertical position of the avatar, the zoom factor, the introduction of a new object by an animation, and so on.
6. In this area the user can control the used memory.
7. In this area the user can control the network status, in particular the messages FIFO queue.

6. REMARKS

The evaluation of the performances of distributed architectures is an extremely critical and hard task. A set of parameters has to

be taken into account; in particular, for a distributed visualization system based on a P2P we have to consider:

- Response times
- Scalability
- Reliability

Different tests have been performed. The first one considers a network involving only peers placed on the same LAN; in particular, a peer is configured as a rendezvous connecting four edge peers. Delays in communications are almost unnoticeable, while reliability problems arise in the event of a rendezvous crash. In this case, the users connected by edge peers are still able to move within the virtual world, but all services (file sharing and so on) supported by XMS (see Section 5) do not work anymore.

The second test involves a configuration where three rendezvous peers are connected in geographically distributed network over the north west of Italy (see Figure 4 the rendezvous in Torino (1) knows the rendezvous in Vercelli (2) and in Ivrea (3), but rendezvous 2 and 3 do not directly know each other. All messages are propagated without loops and, moreover, in the event of a crash of the rendezvous 1, after a timeout delay, rendezvous 2 and 3 are able to begin a direct connection by-passing the rendezvous 1. This example shows how different parts of a complex distributed network can overcome crashes of intermediate rendezvous nodes.

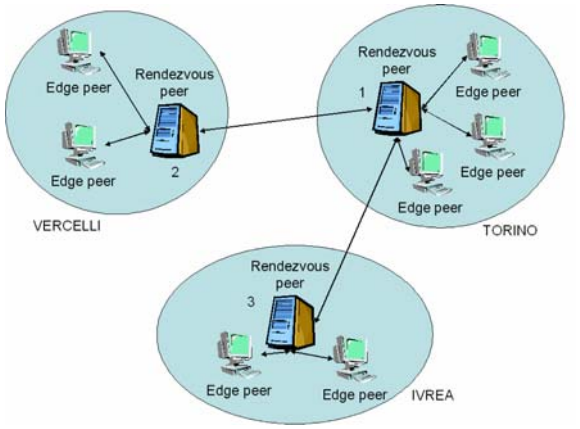

Figure 4: The second test network

The third and last test involves a complex network with many rendezvous and many edge peers. Response times depend on several parameters but, above all, on the bandwidth of network connections. response times due to roto-translation commands can vary from tenths of second to some seconds, while the publication of a model (users have to locally download the model to be able to visualize it), of course, depends on the size of the file to be broadcasted. On the other hand, system reliability is not completely guaranteed by native JXTA fault-tolerance mechanisms.

For this reason, a peer monitoring systems has been developed: each peer sends at regular interval times an *is_alive* message to all the other peers. When a peer crashes, all the other nodes do not receive its *is_alive* message and can reconfigure themselves:

- If the crash affects a rendezvous, all edges peer connected to that rendezvous will find another rendezvous peer that "adopts" them.
- In the event a crash affects the only rendezvous of the network, an edge peers can be elected as new rendezvous by a native JXTA mechanism.

Finally, from the point of view of the scalability, JXTA is designed to provide a support to interconnect hundred thousand peers (more details concerning JXTA performance can be found at: http://bench.jxta.org).

Our tests involved about ten peers and in that scenarios we never experienced any scalability problem.

7. CONCLUSIONS AND FUTURE WORK

The proposed architecture is based on the JXTA technology to implement the distributed network architecture and on the Java3D libraries to visualize and manage 3D objects. Users can meet within the virtual world represented by an avatar; moreover, they can communicate using a chat system directly supported by JXTA.

D3D can be used as a platform to implement services for 3D computer graphics. D3D allows users to share a distributed visualization session where each participant can load and share his/her own models. When a user loads an object can inspect it (all operations performed are translated in messages and sent to the other peers/users that will see the same actions) or release the control.

The proposed architecture is platform independent as all technologies used are available for the main operating systems and hardware configurations; moreover, all software is freeware. The message propagation system based on the DHT Walker guarantees efficiency and scalability, while an ah-hoc mechanism provides the system reliability.

We are currently working to improve the communication mechanism among users. In particular, a videoconference session will be added in order to enhance the interaction capability. Quality of service (QoS) mechanisms will be integrated in the framework; in this way, the quality of the videoconference sessions will be dynamically updated according to a set of parameters such as: available bandwidth, number of participants, local CPU power, and so on.

8. ACKNOWLEDGMENTS

Authors wish to thank ing. Marco Boz for his support in designing and developing the proposed architecture.

9. REFERENCES

[1] S. Androutsellis-Theotokis. A survey of peer-to-peer file sharing technologies. White paper, Athens University of Economics and Business, 2002.

[2] C. S. Ang, D. C. Martin, and M. D. Doyle. Integrated control of distributed volume visualization through the world-wide-web. In *Proceedings of IEEE Visualization' 94*, pages 13–20, 1994.

[3] E.W. Bethel and J. E. Shalf. Cactus and visapult: An ultra-high performance grid-distributed visualization architecture using connectionless protocols. *IEEE Computer Graphics and Applications*, 23(2):51–59,2003.

[4] S. Botros and S.Waterhouse. Search in jxta and other distributed networks. In *Proceedings of the First International Conference on Peer-to-Peer Computing*, pages 30–35, 2001.

[5] I. Clarke, O. Sandberg, and B.Wiley. Freenet: A

distributed anonymous information storage and retrieval system. In *Proceedings of the Workshop on Design Issues in Anonymity and Unobservability*, 2000.

[6] K. Engel, P. Hastreiter, B. Tomandl, K. Eberhardt, and T. Ertl. Combining local and remote visualization techniques for interactive volume rendering in medical applications. In *Proceedings of IEEE Visualization' 00*, pages 449–452, 2000.

[7] N. Frisch and T. Ertl. Embedding visualization software into a simulation environment. In *Proceedings of SCCG 2000*, 2000.

[8] T. Funkhouser. Network topologies for scalable multi-user virtual environments. *Proceedings of VRAIS'96, Santa Clara CA*, pages 222–229, 1996.

[9] L. Gong. Search in jxta and other distributed networks. *IEEE Internet Computing*, 5:30–35, 2001.

[10] J. Heijmans. An introduction to distributed visualization. Technical report.

[11] Rodger Lea, Yasuaki Honda, Kouichi Matsuda, Olof Hagsand, and Mrten Stenius. Issues in the design of a scalable shared virtual environment for the internet. In *Proceedings of HICSS'97*, 1997.

[12] D. T. Lee, S. M. Sheu, and C. F. Shen. Geosheet: A distributed visualization tool for geometric algorithms. *International Journal of Computational Geometry and Applications*, 8:119–156, 1998.

[13] M. R. Macedonia, M. J. Zyda, D. R. Pratt, P. T. Barham, and S. Zeswitz. NPSNET: A network software architecture for large-scale virtual environment. *Presence*, 3(4):265–287, 1994.

[14] Michael R. Macedonia and Michael J. Zyda. A taxonomy for networked virtual environments. *IEEE MultiMedia*, 4(1):48–56, 1997..

[15] R. Minghim, V. C. L. Salvador, B. S. Freitas, M. C. F. Oliveira, and L. G. Nonato. Distributed sound for volumes-data analysis using distributed visualization and sonification. In *Proceedings of SPIE – Visualization and Data Analysis 2002*, volume 4665, pages 379–390, January 2002.

[16] S. Ratnasamy, P. Francis, R. Handley, M.and Karp, and S. Shenker. A scalable content-addressable network. In *ACM SIGCOMM*, pages 161–172, 2001.

[17] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for largescale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms*, 2001.

[18] V. C. L. Salvador, R. Minghim, and H. Levkowitz. DSVOL II - a distributed visualization and sonification application communicating via an xml-based protocol. In *Proceedings of the XV Brazilian Symposium on Computer Graphics and Image Processing*, 2002.

[19] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *ACM SIGCOMM*, pages 149–160, 2001.

[20] C. Zunino, B. Montrucchio, A. Sanna, and C. Demartini. A distributed visualization environment for scientific visualization based on jini technology. In *IEEE Proceedings of SCCG'2001*, pages 95–101, 2001.