

Collaborative Research Within a Sustainable Community: Interactive Multi User VRML and Visualization



Stuart Lovegrove, Ken Brodlie



Abstract

Collaborative working, using computer technology as a medium for results, thoughts and ideas, is a goal that is envisaged will enable users to become more productive, efficient and able. With the advances in computer hardware over recent years, collaborative working and virtual reality systems are available for users upon desktop machines. The involvement of the internet has opened up channels of research aiming to deliver these technologies over very wide area networks and in a manner that moves away from proprietary hardware and software systems. VRML, born from a goal of producing a transport mechanism for three dimensional scenes over the internet, has developed into a system that is now capable of including collaborative, multi user, virtual reality and internet technologies. It is the combination of these technologies that will allow a change in working practices and will provide a work environment for use in areas such as collaborative visualization, shared simulation environments and group design.

This paper discusses ways in which VRML can be used to support multi user shared environments. We introduce current multi user solutions and additionally present our own. The aim is to extend this solution to deliver a persistent shared space where users may work together for research and design using visualization tools.

Keywords: Multi User Environments, Virtual Reality Modelling Language (VRML), Collaborative Working, Visualization, Internet, World Wide Web (WWW)

1. Introduction

'*Collaborative Research Within a Sustainable Community*' is a term used to describe an environment allowing synchronous, or asynchronous, collaboration between a community of users. These sets of users are to be supported in a sustainable environment where data, actions and results (either final or intermediate) will be maintained throughout time. In order to access this collaborative work environment, both 2D and 3D interfaces will be employed. The data sets used as the central application area of research are those output from scientific experiments, measurements and simulations. It is these data sets that undergo visualizations techniques, the resulting output viewable as VRML [1] models.

A central aim of the ongoing research is to investigate what sort of collaborative system is the most applicable, and best suited, to provide this sustainable community. Additionally a solution to the problem of how to integrate such a system with existing technologies (the internet, visualization services, virtual reality devices) must be discovered. The questions that form the basis of research are:

'How can a community of collaborating scientists be supported by a sustainable open architected computer system in a manner which is an enhancement over traditional

working practices'

'What enhancements can be made to current visualization systems, or what possible new systems, such as those running via the World Wide Web and incorporating internet technologies such as Java and VRML, could be put into place in order to improve the ability of users to interactively perform collaborative tasks'

'How can a persistent shared environment be created to effectively store results from users' interactions'

VRML provides a means to encapsulate visual results, associated with a behavioural event model, that allows easy interaction within a three dimensional environment. Extending this inbuilt functionality with the ability to share this environment across a network infrastructure, forms a new avenue of research within the graphics community. It is envisaged that the creation of portable shared environments running across the internet and accessible as a tool on the desktop will radically alter the way we as individuals work and play together. Indeed some large industry players, (Silicon Graphics, Sony, Microsoft for example) have taken a great interest in the ability of VRML and this multi user potential.

This paper will initially discuss briefly the main multi user VRML systems currently available. We then proceed to describe, demonstrate and analyse our own small scale solution for the same task. Introducing a design for our sustainable community follows, motivated from our past experiences with collaborative scientific visualization systems. Finally a brief indication of the future research is illustrated.

2. Multi User VRML Solutions

There are a number of groups attempting to support multi user working with VRML in some manner. Here follows a brief description of each of these methods.

2.1 Living Worlds [2]

The Living Worlds proposal is born out of the desire to produce a generic multi user interface extension to VRML. It has its own working group under the control of the VRML Consortium and is supported by many companies involved in computing, graphics and virtual realities. The aim is to provide a set of extensions to the current VRML specification within the defined syntax. By the development of a defined interface it is hoped that a standard multi user implementation may be added to VRML for future revisions.

The proposal is concerned with the abilities of a system to share objects and pass messages between these objects, the users and the multi user server technology (MUtech). The current interface is deliberately generic to allow any form of collaborative working from within its framework.

This system is to be platform independent by the use of only standardised VRML/Java [3] and a specified prototype interface, allowing connection to a proprietary server. The proposal deliberately does not concern itself with specific details of any server that would pass communications between users.

A multi user system developed with this architecture would involve the addition of prototypes to a VRML world. It is with these prototypes that the methods of the shared environment (object behaviours, communication pathways, security etc.) are encapsulated.

At this stage the proposal is early in its development cycle (currently in the initiation of a second draft). Thus no major demonstrators have yet been produced to illustrate its functionality and its effectiveness in any typical situation. As it stands, the Living Worlds specification offers a flexible, although potentially complex, interface of prototypes that, the

authors claim, will handle complex methods of working within a shared environment.

2.2 Open Communities [4]

The Open Communities proposal is a combined effort from MERL/Chaco/Velocity/Worlds Inc. and like the previous proposal aims to provide extensions to VRML to allow multi user abilities.

The technology builds on Mitsubishi's Spline software and supports a distributed content space across the internet. VRML may be included as part of this content space. The proposal aims to provide an open standard which would act as a layer between the user and any proprietary server technology. The proposal implies that such a standard layer may sit on top of, or replace, Living Worlds.

Connectivity within the shared system is a hybrid of peer to peer networking and client/server technologies. This method of networking infrastructure has been adopted to support an improved level of scalability over a pure client/server approach.

The proposal describes very complex interactions which, if supported by working demonstrators, would prove the solution to be very powerful and suitable in its field. However, there is no evidence of actual demonstrators and there have been no recent updates from the developers.

One criticism that must be levelled at the whole proposal, is lack of evidence of the system's abilities, use or functionality. A high level description of its interconnectivity is provided and an API (Java or C++) is detailed but not of how one would create an actual multi user environment using the system. As such there is little mention of connectivity between the system and the VRML language.

2.3 Sony's Virtual Society Technologies [5]

Sony's browser and server technologies allow VRML to become collaborative. Proprietary extensions to the VRML language via a prototype interface, allow access to extra browser functionality and to Sony's sophisticated server technologies.

The Virtual Society[6][7][8], as the collective group of technologies is referred to, has at its centre an extensively developed server solution. Here users view sections of the worlds, based on their location, and only receive events and messages from other users who are in range of their 'aura'. This aura surrounds each user and defines a zone in which they are able to share space. The aura technology allows users to be spread across multiple servers and increases the scalability and responsiveness of the traditional client server technology on which it is built.

VRML is used as both a geometric file format and a description for Java implemented behaviours within the environment. Connection to the shared space is achieved via a set of prototypes that access specialist functions of the provided world browser. Simple Shared Scripts (SSS) or Application Objects (AO) provide a means for objects in the world to become under the shared control of users.

This solution relies on a proprietary architecture combination of browser and server, and provides a proven (as witnessed in the recent collaboration with BT and the BBC for the Mirror trial) method of multi user abilities and retains the use of VRML as an integral element.

2.4 Blaxxun Interactive [9]

The system proposed by Black Sun Interactive involves the use of a proprietary server connecting to a specialist plugin (PASSPORT) with a VRML browser providing the rendering

capabilities. The connection between the plugin is achieved by the External Authoring Interface (EAI [10]) mechanism of connecting Java and VRML. It is via the plugin that shared messages are passed to the server and to the other users on the system. The server in use shows a property of near linear scalability, in terms of performance against increasing client connections, from tests of up to 5,000 users. This system is one of the few collaborative solutions that can show linear scalability. [11]

Interactivity within the shared space is tailored towards an avatar based chat facility although not restricted to such. It is claimed that any VRML compliant scene may be incorporated, enabling it to become 'shared'. An API is provided that allows access to the multi user server internals and provides abilities to tailor the system functionality. It is unclear as to the abilities within a shared world or how one actually builds a environment using this system.

2.5 DIVE [12]

DIVE is a distributed collaborative virtual reality system from the Swedish Institute of Computer Science (SICS). In this system, VRML 1.0 may be used as a geometric file format whilst DIVE's internal working and scene database handle behaviours and all distributed collaborative facilities.

This solution is a proven technology based upon a highly scalable multi casting system as its means of network connectivity. It offers highly complex interactions within a shared space and application users are permitted access to the full scene hierarchy and persistent scene database.

Although DIVE offers a highly flexible adequate solution for the problem of multi user systems, it and other similar systems, are of little use in any move towards a platform independent internet based technology, due to its reliance of proprietary internal workings.

2.6 GMD [13]

GMD, and in particular Wolfgang Broll [14][15][16][17], have undertaken extensive research in this area and achieved multi user abilities from within VRML 1.0 and 2.0. Their proposals outline a solution whereby extensions of VRML nodes are created which interface via prototypes to a proprietary browser. It is this browser that handles the communications between clients and the server. The reason for this approach is their feeling that VRML2.0 cannot successfully provide an adequate basis for a shared environment.

The research is based around a multicasting system and shows a high level of scalability. It concentrates on providing a useful shared environment and defines those technologies required. In order to improve performance with large numbers of users they describe a virtual world, partitioned into zones. A movable area of interest defines geometry that is visible to the user in that particular zone. Additionally an 'aura' style system, similar to that from Sony, defines a dynamic region in which user communication will occur. If user 'auras' overlap then messages will be passed between them, the content based on the generated interaction.

3. Proposed Multi User Solution

The motivation for providing our own solution comes from the desire to have a working implementation (which at the initiation of the project had not been seen in other proposals) and for it to form the core of our sustainable community. The system has been developed under the following criteria:

- *Aim to build a working solution*

A demonstrator should be built in order to give a proof of concept.

- *Use existing web browsers and internet technology*

The solution shall fit in with existing technologies running through 'standard' tools available on a desktop.

- *Any server will be portable and not reliant on a hardware platform*

Portability of code allows a portability of solution

- *Synchronous collaboration between users shall be supported*

Users shall have the ability to collaborate at the same time.

- *A persistent shared work space shall exist*

Previous states of working should be stored and retrieved at a later date. A persistent shared space allows asynchronous collaboration to occur.

3.1 System Description

Our system is aimed at providing a shared 3D environment in which clients may inhabit and interact. In addition it is tailored towards providing a small scale solution aiming for simultaneous connections of between 1-10 users. This limit is imposed by the intended application of the system (collaborative design and shared working environments for scientific applications rather than large shared 'world' environments) and the use of traditional client server technologies.

The solution is built around the concept of a single controlling VRML file in which a set of nodes (prototypes, sensors, geometry) is included. It is supposed that this VRML file will be dynamically generated when initially requested. A web service would create this file from information held about the scene in an 'environment database'. A standard web browser and VRML plugin is used to provide a view onto the environment with a controlling Java applet running on the browser. On initiation, a client would be presented with a display consisting of three elements:

- A user view on to the environment through which manipulation and navigation may occur.
- A camera view of the environment in which a movable device shows an alternative overview of the environment.
- A 2D shared space aimed at providing additional collaborative interfaces such as, a whiteboard system, a chat system, or other such similar devices.

Together a set of software components form the whole system, with each component written in a platform independent manner. In this way the platform independence of the overall system can be ensured. Inevitably a hardware infrastructure has to be present on which the system must be executed. The software components may be run via browsers which support the combination of VRML and EAI powered Java.

Figure 1 demonstrates the interconnectivity of the various software modules and it is the description of these modules, a demonstration of the system in action and an analysis that form the remainder of this section.

3.1.1 Software Components

The multi user system consists of six elements:

- *The Server*
- *The EngineApplet*
- *The VRML File*
- *The Secondary Camera*
- *Shared 2D Space*
- *The Data Client*

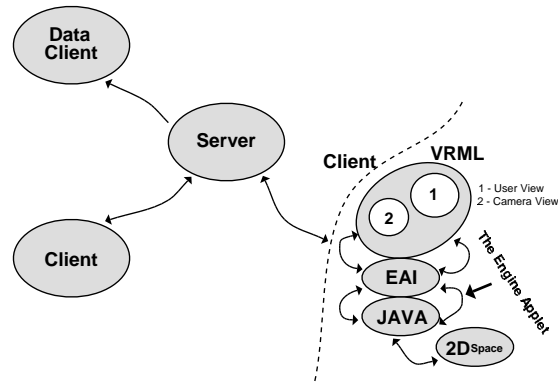


Figure 1: Software Modules

The Server

The central server acts as a message passing service for connected client processes. It is written as a multi threaded application allowing connection requests to be handled whilst maintaining the input/output channels between clients. A broadcasting round robin scheduling system is implemented as the method of passing messages between clients. Messages may either be high level (welcome message) or low level (user position).

Java is used as the language of implementation as it provides a platform independence required by a portable system. Additionally networking and multi threading capabilities are an integral part of this language. As such this eases software development of a server. Communication between threads is achieved via a shared data object, under the control of a monitor, to ensure concurrent access requests may be serviced. Figure 2 shows the abstracted internal workings of the server.

It can be seen that the shared data object is of finite size and places a limit on the number of clients. As more users connect, additional input and output threads are initialised which share a free slot in the data object. When users disconnect from the system, threads and sockets are destroyed or closed and the shared slot is flagged as being reusable. Thus the limited space becomes more flexible.

In addition to passing messages, the server holds the information to provide users with an identification number (currently this ID is determined by the connection order and is an integer).

Incoming messages from user <x> will have the user ID appended before the broadcast stage. This allows the recipients the knowledge of where the message originated and thus the correct action can be taken upon the client side (eg. the correct avatar may be moved).

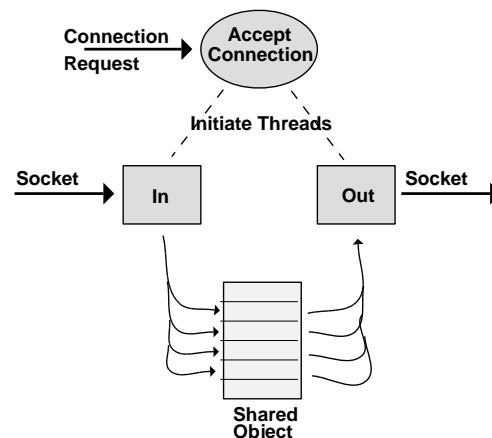


Figure 2: Server Internals

The Engine Applet

This Java applet simply acts as a message passing interface between the VRML world and the network socket connecting to the server. The applet sits on the same page as the VRML world, where it listens out for specific messages being signalled within the world. These messages are picked up, using the facilities provided by the External Authoring

Interface (EAI), and are then packaged into the correct data protocol and transmitted to the server via the network. On receiving a message from the network, it is converted into an appropriate event and passed into the local VRML world. The format of the data protocol is as follows:

```
<message id>|<value 1>|..|<value n>|<user id*>|
```

**NB the user identifier is appended by the server*

The *message id* field holds a unique identifier which determines the semantics and syntax of the remaining elements. For example, <ap> would represent an avatar position with three values whilst <ar> indicates an avatar rotation with a set of 4 values. There is a finite set of message identifiers describing the functionality of the system. (see Table 1)

The applet polls the socket connection to the server under the control of a time sensor node within the local file. The amount of polling can be adjusted by altering the time sensor's attributes. Additionally the applet updates the secondary camera view by passing events from the socket and from the VRML world into the alternate scene. This is a one way flow only, as the secondary view allows no interaction within it other than to move the camera. Details of the secondary camera scene are outlined later.

The VRML File

This file holds all of the geometry to be viewed within the world. Currently all the world geometry is encoded within the scene file as a set of prototype nodes, inlines or straight node descriptions. The file consists of three main elements:

- *The visualization*
- *Proximity and time sensors*
- *A set of external prototype widgets*

Identifier	Functionality
us	new user connection
wc	welcome message from the server
ap	avatar position
ar	avatar rotation
sh	avatar activation/deactivation
pp	shared pointer position
pa	panel setting
ma	marker activation/deactivation
mp	marker position
bu	button activation/deactivation
sc	set scribble point
sd	draw scribble segment

Table 1: Message Identifiers

The visualization is a set of VRML geometry as generated from some external process. Depending on the application of the system, this geometry could be anything from a representation of an architectural model to a graph of the pollutant levels in the last 24 hours. This geometry is currently inlined into the scene and by definition may be any valid VRML file.

A set of sensors in the file controls user tracking and polling of the server socket. A proximity sensor, covering the entire shared space area, allows client positions and orientation to be tracked. A time sensor provides a regular temporal event to allow time sensitive elements to be triggered. Thus the applet can be sent an event causing the socket to be polled for new messages at set intervals. This polling mechanism prevents the overall system being under the exclusive control of the Java applet. On some systems this prevents the VRML browser from allowing interaction and updating the scene.

The external prototype widgets allow shared objects to be collectively used by those on the system. These widgets are again 'hard coded' into the VRML scene although later developments will allow them to be added as and when required. A collection of six widgets have been developed to provide a useful functionality within the environment:

Slider - The slider is a VRML representation of a traditional user interface device and allows the users to alter the value of a variable within the world. An example of this would be when users wish to alter the scale of an object. For example a shared slider may hold the current scale value and may be manipulated by any client to update the scale of the shared visualization.

Pointer - This represents a pointing device constrained within a two dimensional plane inside the shared environment. It allows users to indicate a position of interest within the environment.

Avatar - An avatar is a representation of a user in the scene. Its inclusion is important as it allows participants within system to see where other users are located and what they are viewing. Complexity of the avatar was kept at a minimum and all users are represented in the same manner, a 2D plane with attached texture mapped facial image with 'floating' text indicating user identification.

Marker - The marker, and its associated sensor, surround the visualization geometry to allow an alternative method of indicating the area of interest within the scene. The touch sensor that surrounds the visualization registers the point in three dimensional space of mouse interactions upon the geometry. These 'hit points' and the active status of the sensor are routed to the marker widget, marking the surface with a sphere. This enables clients to see 'touches' upon the geometry.

Swap button - The swap button demonstrates the use of a switch within the environment which may be wired up to activate a specific functionality (eg. switching geometry on or off). The button therefore is a generalised firing mechanism.

Head up display - This device allows other widgets to be collected together and presented to the user. The head up display is permanently situated in front of the users current position. This is achieved by passing the client movement events to the head up display, allowing it to move in tandem.

The Secondary Camera

This device allows an independent view of the world to be observed. The camera is movable in its own separate window and shows all the visible events occurring in the world. Thus the local client, all the additional clients and the shareable widgets may be viewed. In reality the camera is another VRML browser plugin running the same VRML file but connecting only to the engine applet. Events travel down a one way channel to the window, from the applet, to update the view. The benefit of the camera view is the enhancement of navigation within the world. Having two viewpoints in which to see the environment offers more flexibility than a single line of sight. Additionally it would be possible to allow the camera to track the local or selected client. [18]

Shared 2D Space

This space is a shared interface running alongside the VRML browser, presenting an alternative method of communication. The space exists within the engine applet's Java window and acts as a whiteboard for the connected clients. Clients are able to draw shapes on the whiteboard using the mouse, the details of which are shared via the server and the

network. Each client has a local copy of the space and only changed events are transmitted, for example drawing new lines or clearing the space. The whiteboard idea demonstrates the type of 2D application which may be successfully integrated with the multi user system. Others such as shared textual objects, chat systems, multimedia systems and video conferencing could similarly be added.

The Data Client

The Data Client module simply acts as the forerunner of a DBMS system within the current architecture. At this stage of development this module allows data within the network to be streamed off to a standard output device (a window) for inspection. The client in this case is a one way stream with no means to interact with the shared environment.

It is envisaged that the data client module will be developed into a DBMS system with the ability to store events as they are generated and transmitted. By selectively storing these events (eg. storing current avatar locations or a range of locations) the system may persistently store the environment. With a store of the current environment state, along with a record of past results, actions on a temporal basis may occur. Thus backtracking of events and the storage of a world's state for later retrieval can exist. Additionally the server may then be able to use the DBMS as a querying mechanism to enable details about the current world to be ascertained.

3.1.2 Software/Hardware Infrastructure

The current solution uses Java JDK (1.0.2/1.1.2), CosmoPlayer 1.0.2, Netscape 3.01S and a SGI O2 workstation as its development platform. This platform has been chosen as it provides a powerful rendering sub-system coupled with powerful numerical manipulations. All of these components (except the Java JDK) may be substituted with alternative systems. Networking is achieved by standard Ethernet and TCP/IP protocols although the system is shielded from these by the use of the Java networking packages which abstract away from any network specifics.

3.1.3 In Practice

In practice, the system is initiated from a web page which allows some connection choices to be made (location of sever machine and socket port number). A web page with the VRML files(s) and applet is then loaded, and based on the initial choices, the system connects to the server. Figure 3 shows the resulting typical scenario. Once loaded the clients may begin to interact with the environment. This is either by moving and using the shared objects, or the two dimensional space, drawing pictures or annotations. As new users connect, their avatars become visible within the world.

3.1.4 Critical Evaluation

We evaluate our system using the following criteria:

- *Scalability*
- *Interactive Capabilities*
- *Platform Independence*
- *Persistence*
- *Flexibility*
- *Demonstrator*

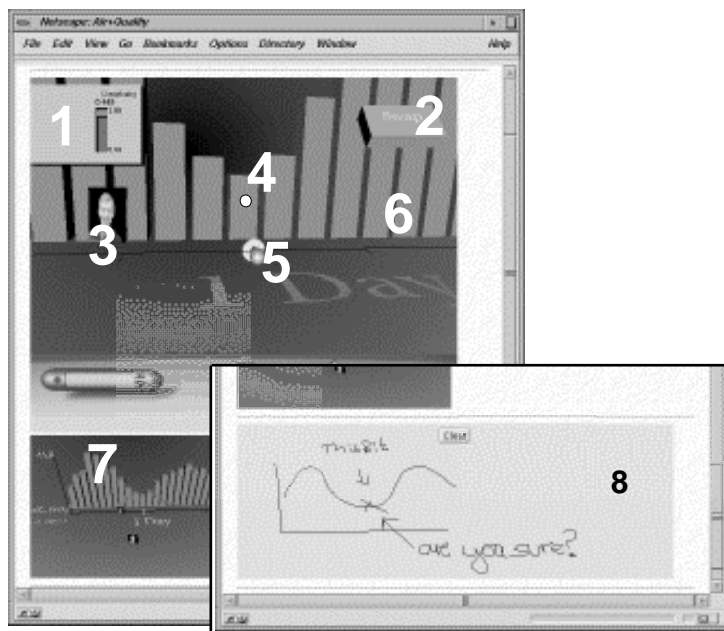


Figure 3: The System

System Elements

- 1 - Slider**
Allows the visualization to be scaled
- 2 - Swap Button**
Replaces the current scene
- 3 - Avatar**
Represents a user in the world
- 4 - Marker**
Allows a point on the visualization surface to be indicated
- 5 - Pointer**
Allows a method for "pointing"
- 6 - Visualization**
The processed data set
- 7 - Secondary Camera**
Allows an alternate view onto the world
- 8 - 2D Shared Space**
Demonstrating a whiteboard

The system has poor scalability caused by the poor scalability of the client server approach. Increasing the number of users within the system increases the data flow through the central server resulting in a bottleneck. However, within the proposed application area, that of small scale scientific collaborative systems, the maximum number of clients (1-10) is more than adequate. Within this area the amount of users working together on a project is likely to be that of small groups. With high numbers of people, overall user productivity is likely to decline.

The use of Java and VRML allows portability and independence. Additional requirements of the system are that of a VRML browser, and HTML browser and the EAI class mechanisms for Java/VRML interoperativity.

Within the range of objects provided, interactive capabilities are well supported and allow a large range of activities within the environment. However, interactions are only provided for a finite set of objects. Thus the overall system has a restricted interactive capability.

Persistent scenes, data and results are not currently stored although the data client will evolve into a module that can handle this facility. Only the current world state is held; although not specifically stored it can be retrieved by querying connected clients.

Within the limited domain of the system the issue of flexibility is addressed adequately. The range of supplied tools allows a useful range of actions to be taken. As an overall system, the lack of ability to add objects to the scene reduces the global flexibility.

The presence of a demonstrator shows that the development of a multi user system using these techniques is feasible in certain applications (eg. small client load, limited flexibility etc).

In response to this analysis it must be noted that the system is currently at early stages of development and is presented here as a proof of concept system. It is from the above criticisms that the future technical work is motivated. Issues regarding flexibility and

interactivity of the scene may be solved by the creation of a generic shared object module. This shared object module would possess a generic input output mechanism for message passing and contain its own behaviour.

4. Application to Visualization

The goal of the development of our multi user system is to apply these techniques to the realm of visualization applications. The aim is that by combining the two, a collaborative visualization system will result which will offer users an enhanced method of working and allow support for our sustainable community. The justification for this combination is that a collaborative system will allow:

Better sharing of results - Results may be shared across users' desktops, networks and platforms. The increased platform independence of a WWW based system allows results to be easily propagated. Single user systems have the difficulty of how to share the results of a visualization with other users and how those users can contribute to the process of creation.

Persistent storage of results - Persistent storage of results enables past experimentation to be stored for later use. A persistent state allows the current visualization to be recalled and altered via asynchronous means (users do not have to work together at the same time).

Synchronous working to allow real time feedback - The ability to work in a real time environment where changes are reflected to all connected users allows client actions to be correctly modelled. It allows a form of working that may be more natural.

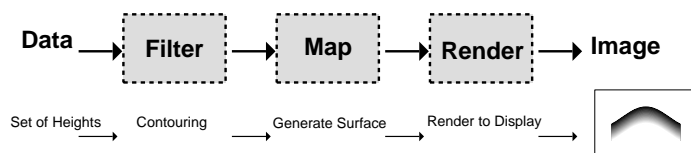
Asynchronous working to allow a temporal spread - Supporting an asynchronous method of working within a persistent environment allows collaboration to be spread over time. This is required if clients are unable to work together within the same time frame.

Ability to interact with the resulting visualization - Interacting with the visualization, for feedback into the visualization pipeline, allows a flexibility of working which may not be afforded if a client is only permitted manipulation before the rendering stage.

The changes brought by moving towards a collaborative system may be seen to alter the traditional visualization pipeline as defined by Haber and Mc Nabb. [19] We propose an addition to the standard pipeline in Figure 4. Here it can be seen that interactions from the clients feed either into the rendering process or directly back into the visualization process. Thus the pipeline may be spread across to the client allowing them to undertake more of the actual processing.

The consequences of developing such a system are:

Original Pipeline



Altered Pipeline

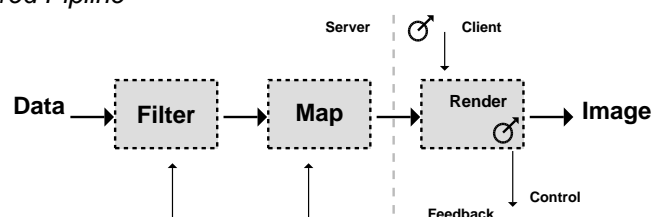


Figure 4: Visualization Pipelines

Work may be carried out by WWW tools - The GUI front end of the visualization engine maybe provided via Java, through a standard WWW browser. This increases the flexibility of the software in terms of platform independence.

Sharing the visualization pipeline - The load of the pipeline may be spread across multiple machines easily (eg. the main visualization process occurs on machine A, whilst the rendering occurs on machine B).

4.1 Collaborative Visualization System Design

The design of the collaborative visualization system is motivated from previous work in this area. The COVISA and WWW Visualization projects both offer a solution to the problem of providing a collaborative working environment where information maybe exchanged between users. It is the concepts and ideas from this work that will feed research into the support of a sustainable community.

4.1.1 Previous work

COVISA - The COVISA project [20] investigated collaborative scientific visualization extensions to the IRIS Explorer [21] visualization software. Work was successfully completed in this area, the result of which offers IRIS Explorer users the ability to work collaboratively, and synchronously, within the usual IRIS Explorer desktop interface. IRIS Explorer's basis is the setting up of a visualization pipeline consisting of modules and the connections between them. Collaboration occurs via set of shared modules within this interface, through which the data flow may be routed across machines and between users. A flow is set up to and from the shared nodes and as such, data values may be transmitted between users via a specialist server. COVISA offers synchronous working upon specific platforms (IRIX and Windows) and operates using a standard sockets interface to facilitate data transfer across networks. Figure 5 demonstrates the routing that occurs within the system.

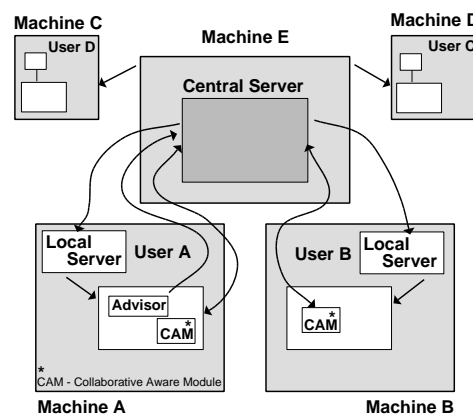


Figure 5: COVISA

WWW Visualization - The WWW based visualization project [22] forms an asynchronously collaborative system that uses a web based forms interface to control parameters of a visualization pipeline to provide rendered results (using VRML version 1.0 or 2.0). Figure 6 shows the process involved in the system.

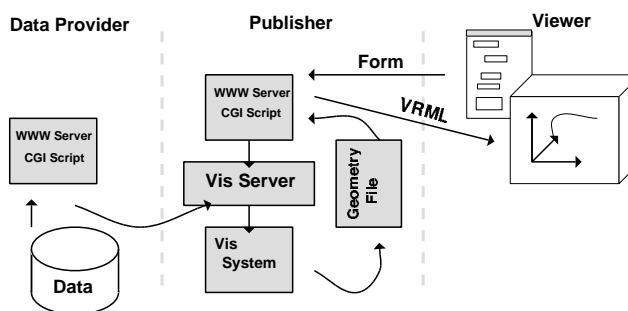


Figure 6 - WWW Visualization

The forms interface allows variables within the pipeline to be altered and, when this information is submitted to the web service, a remote visualization engine acts these variables and a rendered result is created. The rendered output forms a set of VRML geometry which is returned to the user

for viewing in a local VRML viewer.

Collaboration between users occurs via the retrieval of past visualization through a tree structure. This tree structure holds information about past results in the system and allows a structured way of organising sessions and results. The tree may be annotated which provides the group members a means to work together on the tree. Thus asynchronous working may be supported.

4.1.2 New Design

Presented here is a proof of concept design showing a sample multi user visualization system, with interactively collaborative features, and a platform independent implementation. A global system has been envisaged and is shown in Figure 7. Here we can see the various elements of the system allowing a visualization and the support of its client community to take place.

The various aspects of this design are as follows:

Data Provider - This is a service which will provide a set of data as required. It is this data that is to be visualized. The data provider may be a repository of data from another process or may itself be a process (eg. some part of a simulation or design application).

Visualization Service - This is a service that will take the data and put it under some set of operations, the result of which will be a collection of geometry in VRML format. (eg. a graph, 3D surface plot etc).

Web Service - This is a service which will handle HTML requests from the users. It will allow users to initiate the visualization service to produce VRML from the data provider. The web service allows the users a connection to the multi user server where the environment may be shared. In this description the web service is described as a separate entity, in practice it would be feasible to combine this within the multi user server.

Multi User Server - This module provides a 'hub' which will act as a server connected clients. It is simply a message passing service receiving inputs and passing them the clients. The multi user server could also connects to other non client services. The reasons for this connection are:

- One type of interaction within a scene would be to alter parameters originally input as part of the visualization process. These instructions pass to the server, are routed to the visualization service, to provide a new set of geometry. This new geometry may then update the current environment.

- The multi user server will have the ability to connect to the DBMS service which holds the state of the environment at a specific time. When the world is requested in another

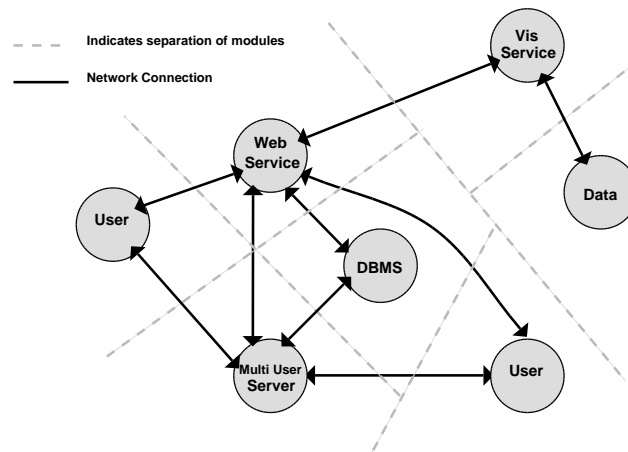


Figure 7: General Overview

session, previous interactions by users would be recalled from the DBMS service and fed into the generation process, altering the file returned to the user.

DBMS service - This service provides a storage facility and associated database management system which will store the messages arriving at the multi user server. This service will also have the ability to return previous messages to allow backtracking of events. The DBMS facilitates the idea of separate sessions where different users could carry out alternate interactions with the scene. Additionally the DBMS could store past visualizations from previous sessions. This would enable past results to be easily recalled for further use.

Client - The client refers to the user operating their machine with associated web browser and VRML viewer. The user will request a visualization from the web service and provide the location of a suitable data set. This will be sent to the the web service which will return a VRML file. The user will run this VRML file locally on their machine. This action will connect the VRML visualization to the multi user server.

4.2 In Practice

In practice a typical session using the system might proceed as follows:

Visualization Of Weather Data - Client A loads up a web browser on his desktop computer and accesses the weather data visualization page. After entering a password, a depiction of previous visualizations within his group appears, allowing a selection of last week's weather data. Client A decides to opt for the latest weather data and sees that client B is already in that shared environment. Client A selects some tools he wishes to bring to the environment and hits the submit button. The web server responds by building a VRML file, depending on the state held in the DBMS, and passing it back to his machine. The VRML browser is then activated and the Java applet loaded which connects to the multi user server.

On connection to the shared environment, A sees user B's location on his camera overview and that she has set up an interesting environment state. Client A navigates to her position using his main controlling window and sees she (B) has left some pointing devices around the environment to indicate areas of interest. Client A is interested in the weather conditions in the early part of the morning but is unsure which part of the visualization represents this data. He moves to use the 2D shared environment and writes a message for Client B to ask her for her help. Client B duly obliges and moves the pointer device to the area that A is interested in.

Thus the session continues and Clients A and B work together to gain an insight into the week's weather data using the shared tools available. The DBMS system keeps a track of all their movements in the current session to allow them to backtrack, to the point when the session was deemed to have begun, though the past events. When the connected clients are happy with their work the server may be informed the session has ended which prompts the DBMS to make a record of positions, pointing devices, annotations etc. for later retrieval. Users A and B then leave, ready to receive the data for the following week!

5. Future Work and Conclusions

5.1 Future Work

For the attainment of our goal of a '*collaborative research within a sustainable community*' we shall be looking to combine the multi user core with our global collaborative design.

In the multi user VRML system area, the following will become the basis of future technical work:

Improve flexibility - There is a need to provide a method of generic objects to improve the flexibility of scene creation. As seen in the analysis the current system has limited flexibility due to the reliance upon a previous defined VRML file.

Adaptive controls - There is a need for the ability to influence the rendering/visualization pipeline from within the VRML scene. Provision of a method to represent selected parts of the visualization pipeline within the shared environment may be required.

DBMS - Persistence within the shared environment requires the addition of the DBMS module element.

Improved Interaction Capabilities - There may be a need to provide additional devices within the world and to be able to control the environment from external sources other than the mouse (eg. video tracking, space balls)

Viewpoint Sharing - Ability to join views together into a shared movable viewpoint, or to jump to the position of another user, allows an improved method of navigation and information sharing,

Camera Tracking - An ability to track user positions by assigning the camera to their avatar, allows the camera model to offer a greater range of functionalities.

Java behaviour Model - The current system uses the EAI method of Java/VRML connectivity. This forces all Java behaviours through a single applet upon a web page. Supporting the internal scripting Java interface allows behaviours to be defined as part of the VRML object concerned. This aids flexibility, object orientation and data hiding.

Server Improvements - Improving the server model (eg. multiple servers, auras etc.) would alleviate bottlenecks, improve scalability and enhance overall system responsiveness. Alternatively a supporting peer to peer networking and multicasting message passing could be introduced to reduce the reliance on a central server.

Increasing Client Load - More of the processing element of the visualization pipeline could be moved to the client. These 'thick' clients effectively spread the system load across the connected machines. Also this attempts to move traditional visualization 'engines' into a web based environment.

5.2 Conclusions

In conclusion it can be seen that the proposed multi user solution can stand as an alternative to the other systems. Whilst it lacks a certain amount of flexibility and maturity found in the other developments, it demonstrates that VRML is capable of supporting such an environment. With the standard node set and the presence of Java collaborative VRML is feasible. The global collaborative design aims to take this multi user core and incorporate it and other technologies to support a collaborative working environment.

6. Acknowledgements

We wish to thank Dr. Ken Brodlie for his presence, support and encouragement throughout this research.

7. References

- [1] VRML '97 Specification, <http://vag.vrml.org/VRML97 DIS/>
- [2] Living Worlds Specification, <http://www.livingworlds.com/>
- [3] The Java Language <http://java.sun.com/>
- [4] Open Communities Specification, <http://www.merl.com/opencom/>
- [5] Sony, Virtual Society, <http://www.csl.sony.co.jp/project/vs/>
- [6] Lea R, Honda Y, Matsuda K, "Virtual Society: Collaboration in 3D spaces on the internet", 1996
- [7] Lea R, Honda Y, Matsuda K, "Issues in the Design of a scalable Shared Virtual Environment for the internet", 1996
- [8] Lea R, Honda Y, Matsuda K, Matsuda S, "Community Place: Architecture and Performance"
- [9] Blaxxun Specification, <http://www.blaxxun.com/>
- [10] The External Authoring Interface, <http://cosmo.sgi.com/>
- [11] Blaxxun Linear Server Performance, <http://www.blaxxun.com/products/server/docs/scale.html>
- [12] Distributed interactive Virtual Environment, <http://www.sics.se/dce/dive>
- [13] German National Research Centre for Information Technology, <http://orgwis.gmd.de/>
- [14] Broll W, "Interaction and Behaviour in Web Based Shared Virtual Environments", IEEE Global Internet 1996
- [15] Broll W, "VRML and the Web: A Basis for Multi User Virtual Environments on the Internet", 1996
- [16] Broll W, England D "Bringing Worlds Together : adding Multi-User Support to VRML" VRML 95, 1995
- [17] Broll W "Distributed Virtual Reality for everyone - a framework for Networked VR on the internet", IEEE VRAIS 97
- [18] Edwards J, Hand C, "Maps: Movement and Planning Support for Navigation in an Immersive VRML Browser"
- [19] Haber R, McNabb D "visualization Idioms: A Conceptual Model for Scientific Visualization Systems", IEEE 1990
- [20] Wood J, Wright H, Brodlie K "CSCV - Computer Supported Collaborative Visualization", BCS Displays Group International Conference on Visualization and Modelling, 1995
- [21] IRIS Explorer 20, Technical Report, Silicon Graphics Computer System, Mountain View 1992
- [22] Wood J. PhD Thesis, University of Leeds 1997