BINDING SERVICES
Tel +44 (0)29 2087 4949
Fax +44 (0)29 20371921
e-mail bindery@cardiff.ac.uk

VICE

An Interface Designed for Complex Engineering Software: An Application of Virtual Reality

Thesis submitted for the
Degree of Doctor of Philosophy
for the University of Wales,
Cardiff

by
Mark John Taylor
(MEng)

September, 2005

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Declaration

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed..... (candidate)

Date 20/3/06

STATEMENT 1

This thesis is the result of my own investigations, except where otherwise stated.

Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed..... (candidate)

Date 20/3/06

STATEMENT 2

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed..... (candidate)

Date 20/3/06

Abstract

Concurrent Engineering has been taking place within the manufacturing industry for many years whereas the construction industry has until recently continued using the 'over the wall' approach where each task is completed before the next began. For real concurrent engineering in construction to take place there needs to be true collaborative working between client representatives, construction professionals, suppliers and subcontractors.

The aim of this study was to design, develop and test a new style of user interface which promotes a more intuitive form of interaction than the standard desktop metaphor based interface. This new interface has been designed as an alternative for the default interface of the INTEGRA system and must also promote enhanced user collaboration. By choosing alternative metaphors that are more obvious to the user it is postulated that it should be possible for such an interface to be developed.

Specific objectives were set that would allow the project aim to be fulfilled. These objectives are outlined below:

- To gain a better understanding of the requirements of successful concurrent engineering particularly at the conceptual design phase.
- To complete a thorough review of the current interfaces had to take place including any guidelines on how to create a "good user interface".
- To experience many of the collaboration systems available today so that an informed choice of application can be made.
- To learn the relevant skills required to design, produce and implement the interface of choice.
- To perform a user evaluation of the finished user interface to improve overall usability and further streamline the concurrent conceptual design.

The user interface developed used a virtual reality environment to create a metaphor of an office building. Project members could then coexist and interact within the building promoting collaboration and at the same time have access to the remaining INTEGRA tools. The user evaluation proved that the Virtual Integrated Collaborative Environment

(VICE) user interface was a successful addition to the INTEGRA system. The system was evaluated by a substantial number of different users which validates this finding. The user evaluation also provided positive results from two different demographics concluding that the system was easy, intuitive to use with the necessary functionality.

Using metaphor based user interfaces is not a new concept. It has become standard practise for most software developers. There are arguments for and against these types of user interfaces. Some advanced users will argue that having such an interface limits their ability to make full use of the applications. However the majority of users do not come within this bracket and for them, metaphor based user interfaces are very useful. This is again evident from the user evaluation.

Dedication

I dedicate this thesis firstly to my parents, Peter and Angela Taylor for their never ending love and support. Secondly to Lindsay, someone very special in my life who has made this whole experience more bearable.

Acknowledgements

I would like to take this opportunity to express my gratitude to my supervisors Professor John Miles from Cardiff University and Professors Chimay Anumba & Dino Bouchlaghem from Loughborough University for their continual assistance and guidance throughout this study.

Special thanks to David and Lisa for their support and to my housemates Andy, Rachel and Anna for their help during some of the tough times. Also thanks to Carole and Ben and the entire Nunn family who have often given me much needed encouragement.

Finally, I would like to express a deep gratitude to the rest of my friends particularly my water polo family, both from Cardiff University and Cheltenham. Many of them thought I was mad to stay at University for so long, but without them I probably wouldn't have!

1 Introduction

2 Concurrent Engineering & Collaboration Systems

3 Graphical User Interface Design & Metaphors

4 Virtual Reality (VR) & VRML

5 Creating VICE

6 INTEGRA VICE

7 User Evaluation

8 Conclusions

Bibliography

Chapter 6 – INTEGRA VICE

Chapter 7 – User Evaluation

CHAPTER 1

Introduction

1.1 Aim of the Research

The aim of this research project has been to design, develop and test a new style of user interface, which promotes a more intuitive form of interaction than the standard desktop metaphor based interface. This new interface has been designed as an enhancement and alternative for the default interface of a collaboration system know as INTEGRA . By choosing alternative metaphors that are more obvious to the user it is postulated that it should be possible for such an interface to be developed. INTEGRA is explained in detail within chapter 5, but it is basically an internet-based software system that supports the concurrent conceptual design of commercial buildings.

Concurrent Engineering has been taking place within the manufacturing industry for many years whereas the construction industry has until recently continued using the 'over the wall' approach where each task is completed before the next began. For real concurrent engineering in construction to take place there needs to be true collaborative working between client representatives, construction professionals, suppliers and subcontractors. This collaboration can be achieved without IT through co-location. However, co-location is expensive, time consuming and impractical. It normally involves a lot of travel, and often causes disruption amongst construction projects. The information technology required to facilitate this new collaboration without co-location is now available. The barriers that are stopping this introduction of technology and efficient effective collaboration are human and organisational.

It is during the conceptual design phase that concurrent engineering has a clear effect when considering basic building construction. Most of the cost of a project is determined by decisions made early on in the design stage of a project, usually during the conceptual design stage. The 80/20 rule is often referenced, which states that 80%

of a project's cost is determined by decisions made within the first 20% of the project effort. Many engineers now believe that this ratio is accurate enough with some predicting a ratio nearer 95/5. Needless to say conceptual design is a challenging area which needs the development of new and different techniques to help designers rapidly assess and develop their new ideas (Miles et al, 2001).

For this reason INTEGRA chose to look specifically at the conceptual design phase and set about implementing an IT system that would aid the conceptual design process, improving and streamlining the procedure.

The INTEGRA system uses a common frame based interface and standard menu system as its default user interface. To fulfil the initial project aim of creating an alternative metaphor based interface for INTEGRA certain objectives had to be achieved:

- To gain a better understanding of the requirements of successful concurrent engineering particularly at the conceptual design phase.
- A thorough review of the current interface technologies had to take place including any guidelines on how to create a "good user interface".
- To experience many of the collaboration systems available today so that an informed choice of application can be made.
- To perform a user evaluation of the finished user interface to improve overall usability and further streamline the concurrent conceptual design.
- Completing all of the above objections will ensure that the overall objection of creating an entirely new style of user interface can be achieved.

1.2 Arrangement of the Thesis

The seven remaining chapters of this thesis are arranged as follows:

Chapter 2 Concurrent Engineering & Collaboration Systems: The first section of the chapter looks at concurrent engineering, defining what exactly it is and why it is necessary for good conceptual design. This section gives a better understanding of the requirements of successful concurrent engineering in conceptual design. The second

section then looks at collaboration systems, explaining what they are, how they work and why they are used to aid the concurrent conceptual design. It considers various existing collaboration systems and uses those systems to get a better understanding of what is required of a good collaboration system.

Chapter 3 Graphical User Interface Design & Metaphors: This chapter looks at graphical user interface design and how to create a good user interface for an IT system. The first section of the chapter starts by looking at the history of the user interface so that a better understanding can be gained. In addition, what makes a good or bad interface is discussed before explaining the guidelines on creating a good user interface. The second section of the chapter looks in more detail at metaphors an important aspect of modern user interfaces and discusses how they are used when creating user interfaces. Learning the relevant skills required to create a successful user interface was paramount to achieving the objectives.

Chapter 4 Virtual Reality (VR) & VRML: This chapter comprises the specific knowledge and programming skills needed to write the user interface of choice. The chapter starts by explaining virtual reality before examining a variety of virtual environments, both immersive and non-immersive as well as distributed environments. The second part of this chapter examines VRML, the programming language chosen to create the virtual worlds. It describes the specification of the language and looks at how the language is written and compiled.

Chapter 5 Creating VICE: This chapter describes how the knowledge gained in the previous chapters was used to make informed choices on how to create the VICE system. It examined the considerations made before the final selections were made and the interface written. The chapter starts by fully describing INTEGRA, the system for which the interface is being designed. It then considers how the virtual environment will be modelled and distributed for multiple users. Finally the chapter considers the various options for communication, required for the collaboration that will allow successful concurrent engineering.

Chapter 6 INTEGRA VICE: This chapter describes the INTEGRA VICE system. It contains many screenshots to illustrate the interface and explain all of the tools that can

be interacted with and used. The chapter explains what and why choices were made. The chapter also contains snippets of code.

Chapter 7 User Evaluation: This chapter explains the user evaluation carried out to ensure maximum usability of the system and gain constructive feedback on different types of people's usage of the system. The chapter first explains why user evaluations are needed in interface design and then describes the two evaluations carried out. The chapter also analyses the results from the evaluations.

Chapter 8 Conclusions: This concludes the thesis by looking at the successes and failures of the created interface and considering work which could be carried out in the future.

CHAPTER 2

Concurrent Engineering & Collaboration Systems

The first section of the chapter looks at concurrent engineering, defining what exactly it is and why it is necessary for good conceptual design. This section gives a better understanding of the requirements of successful concurrent engineering in conceptual design as required in the objectives. The second section then looks at collaboration systems, explaining what they are, how they work and why they are used to aid the concurrent conceptual design. It considers various existing collaboration systems and uses those systems to get a better understanding of what is required of a good collaboration system.

2.1 Concurrent Engineering

2.1.1 What is Concurrent Engineering?

2.1.1.1 Dictionary Definitions

"Concurrent: Running in the same direction as parallel lines; (meeting at, or tending towards the same point); existing or acting together or at the same time; agreeing."

"Engineering (n): the application of scientific knowledge to the design, building and use of machines, roads bridges and or electrical equipment."
(Oxford Dictionary 1994)

2.1.1.2 Industry Definitions

The widely accepted definition of concurrent engineering (simultaneous engineering) with regards to industry was the result of a five year study by the Institute for Defence Analyses which produced a report on Concurrent Engineering (Winner et al, 1988). They define concurrent engineering as "a systematic approach to the integrated,

concurrent design of products and their related processes, including manufacture and support. This approach is intended to cause the developers, from the outset, to consider all elements of the product life-cycle from conception through disposal, including quality, cost, schedule and user requirements."

Gillen & Fitz (1991) wrote that "Concurrent Engineering is a systematic approach to the integrated design of products and their related processes, including manufacturing and support."

Natale (1993) of Sun Microsystems defined it as: "Concurrent Engineering is a cross-functional inter-disciplinary activity that begins at the pre-natal stages of design and continues through production and product end of life"

During a lecture at Tufts University in Massachusetts, 1993, Concurrent Engineering, or Concurrent Product Development-CPD, was defined as being an improvement initiative that is focused on reengineering the product development function for speed, efficiency, and quality.

Prasad (1996) states that concurrent engineering is a systematic approach for considering management of a products life cycle that includes the integration of planning, design, production and related phases.

So concurrent engineering is defined as a process that can be implemented in the design and creation of products. It is not a 'product', it can not be bought off-the-shelf. There is also no strict process that can be adhered to when dealing with concurrent engineering. Perhaps concurrent engineering is best described as a management philosophy - a way of thinking about and approaching a situation. To get a better definition of concurrent engineering we need to combine the two definitions. Briggs (1996) suggested the following hybrid definition "Aspects of scientific knowledge running and working in parallel lines towards a given point or goal." The common threads that exist within these definitions are that events and activities will occur at the same time, that the concept of time is dynamic in that it passes, and that a concurrent engineering activity requires a goal. Graphically, this may be shown as in Figure 2.1,

which illustrates this philosophy and shows the parallel nature of the concurrent engineering process.

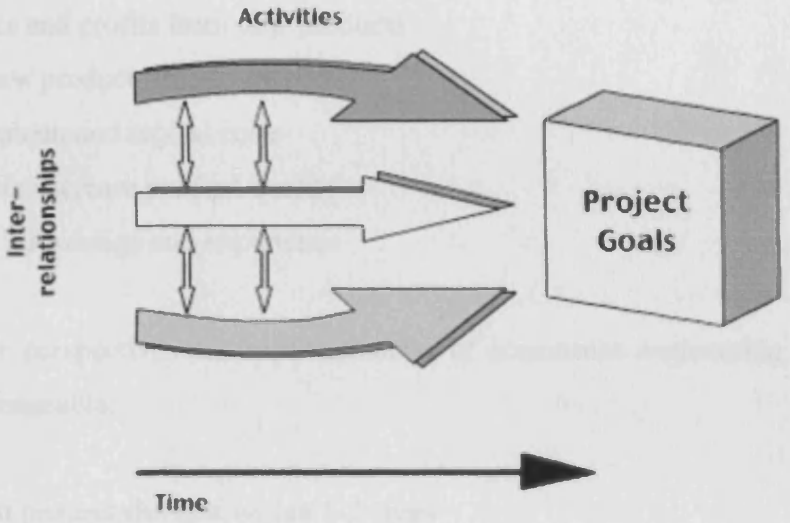

Figure 2.1 – Concurrent Engineering Process

Although originally developed for the manufacturing and product development industry, concurrent engineering has become important to the AEC industry. A more comprehensible definition of concurrent engineering in respect to the AEC industry was comprised by Evbuomwan & Anumba (1998). They stated that: 'Concurrent engineering attempts to optimise the design of the project and its construction process to achieve reduced lead times, and improved quality and cost by the integration of design, fabrication, construction and erection activities and by maximising concurrency and collaboration on working practises'.

2.1.2 The Objective of Concurrent Engineering

The Institute of Concurrent Engineering states that concurrent engineering is used widely by companies such as IBM, NASA and XEROX, resulting in greatly reduced product development time, improved product quality and minimisation of design changes. The National Institute of Standards and Technology (NIST) published that the benefits of concurrent engineering included 30% to 70% less development time, 65% to 90% fewer engineering changes, 20% to 90% less time to market, 200% to 600% higher quality, and 20% to 110% higher white collar productivity. (NIST, 1990)

From an organization-wide viewpoint, concurrent engineering is an "island of change" which generally means that it is for ever changing and hard to define. From a management standpoint, concurrent engineering is definable. The goals are quite clear:

- Raise sales and profits from new products
- Reduce new product time-to-market
- Reduce human and capital costs
- Maintain or increase product quality
- Leverage knowledge and experience

From a scope perspective, the implementation of concurrent engineering programs is finite and manageable:

- Implement process changes within 1-2 years
- Involve people with stakes in new products
- Focus on business process improvements

If these three basic views of project scope are followed, concurrent product development efforts will yield the expected benefits within the planned time period. Many companies focus on technology-based solutions to problems. Almost by definition, the development, implementation, and training cycles for technology-based solutions will exceed 2 years (Goldenese, 1993).

When considering the AEC industry the traditional product development process used has inherent problems and these have been identified by Evbuomwan & Anumba (1996). This sequence is often referred to as the 'over the wall' approach, more of a relay race where processes start after the completion of the previous process. The main disadvantage is the poor communication between all the players involved in the process. Other disadvantages include:

- Elimination of viable design alternatives due to pressure of time;
- Characterisation of the design process with rigid sequence of activities;
- Constructability and supporting issues are considered late in the process;

- Fragmentation of design and construction data and difficulty in maintaining data consistency;
- The occurrence of costly design changes and unnecessary liability claims;
- Loss of information about design rationale and intent; and
- Inappropriate estimation of construction costs

The Latham report (1994), followed by the Egan report (1998) have provided the catalysts for change within the AEC industry. Latham criticised the established industry practises and techniques, concluding that savings of up to 30% can be made if techniques such as concurrent engineering, just-in-time supply, customer supplier relationships and Total Quality Management were introduced into the construction sector. He concluded that there was an urgent need for a reappraisal of procurement and contractual relationships, to create a more open industry that is more willing to share information, when in collaboration, for the benefit of the industry as a whole.

Then in 1998 Egan identified a deep concern that the industry as a whole was under-achieving! He stated that it had low profitability and invests too little in capital, research and development, and training. He also showed concern that too many of the industry's clients were dissatisfied with its overall performance. Within the report Egan outlined the five key drivers for change taken from the key factors behind the manufacturing renaissance in the UK. These factors were:

- Committed leadership;
- Focus on the customer;
- Integrated processes and teams;
- Quality driven agenda;
- Commitment to people. (Egan, 1998)

Egan went on to state that ambitious targets and effective measurement of performance is essential. This could lead to a 10% reduction in construction costs and time and a reduction of 20% of defects.

2.1.3 Problems Implementing Concurrent Engineering

Cleetus (1992) identified many of the problems that occur when attempting to implement concurrent engineering. They are usually a result of any organisational, operational or cultural changes made within a company. The most difficult changes to manage are the cultural changes in an organisation. For a company to successfully implement concurrent engineering they have to fully accept the concept of parallel working. This is often difficult for older companies who have adopted the traditional hierarchical pyramid system for many years. Allowing parallel working amongst employees means that employees of different levels and expertise, often at different levels of the hierarchical pyramid must work closely together to arrive at decisions. A company manager may have to work closely with his/her employee to come to a final decision. Because of the traditional system that the company previously adopted, the employee would be very reluctant to express his/her own opinion, especially if it differed to that of his/her managers'.

Eventually parallel working will experience a problem of consistency. This is because the partitioning of tasks along roles is approximate, an overlap often remains. This results in a conflict among the individual decisions as the parallel tasks progress in time. The problem can be reduced by regular points of synchronisation where emerging alternatives and details of the decisions can be relayed to the entire project group. (Cleetus, 1992)

Another common problem when adopting concurrent engineering strategies arises due to employee's reluctance for change, particularly middle management level. This reluctance is due to:

- No understanding of the need for change
- No experience with the change process
- Natural human tendency to not change (fear of the unknown)
- New processes still subject to old or unknown measurements (Fisher, 1993)

Some employees can not accept concurrent engineering because they can not relate it directly to their job. To overcome this problem teams must be trained to implement

concurrent engineering practises in areas applicable to their assignments. Fisher (1993) also states that training alone is not always sufficient to achieve a concurrent engineering cultural transformation. Every individual involved in product development should have a role to play in the transformation process. Change can not simply ordered by senior management. It must be built by grass-roots involvement, with each individual holding a key role in the change process and measurable progress objectives to achieve. Attempts to implement such changes often fail because although the strategy is clearly understood by the employees implementing it, the individuals who would support and follow this strategy were unaware of its principals and benefits.

Concurrent engineering is all about teamwork. For successful teamwork to occur there has to be good communication. Any lack of effective communication between colleagues will lead to poor team performance. It is this area that computing technology excels at and collaboration technologies must be incorporated to enable effective communication. Sections 2.5 onwards look at collaboration systems in more detail. There are four main areas in which computers can be used to support concurrent engineering: sharing information, collocating people and programs, integrating tools and services with frameworks, and coordinating the team.

2.1.4 Concurrent Engineering – Case Study

All of the information for this section is from the Glaxo Wellcome facility case study featured in the Proceedings of the Institution of Civil Engineers: Structures and buildings, volume 128. (Harryott et al, 1998a and Harryott et al, 1998b)

Figure 2.2 – Glaxo Wellcome facility at Stevenage

2.1.4.1 Introduction

Considering the following case study will allow a better understanding of the advantages and disadvantages of concurrent engineering, specifically within the AEC industry. In 1998 Harryott et al published a report on the design of the Glaxo facility built in Stevenage shown in figure 2.2. The building was designed to contain the most sophisticated laboratories, where the maximum efficiency could be coupled with the maximum safety.

Glaxo appointed the Kling Lindquist Partnership (TKLP) of Philadelphia to prepare the master plan of the entire facility, whilst also selecting the principal architect engineer and the principal contractor to design and build the project. The principal engineer was Ove Arup and Partners, and the architects were Sheppard Robson Architects. Davis Langdon & Everest were the cost consultants for the project. Finally, the principal contractor comprised of Lange Management limited and MK Ferguson of Cleveland, Ohio.

The following sections show how the Glaxo project management team attempted to complete the project using the fundamentals defined by concurrent engineering, in order to successfully complete the project on time, and within the initial budget.

2.1.4.2 Group Organisation

As previously stated, one of the principles of concurrent engineering is to have efficient teamwork throughout any project. From the beginning of this project Glaxo used a simple organisational structure where its project management team, principal architect engineer and principal contractor would work together to produce the building as safely and efficiently as possible. The organisation was built on a triangulated relationship where each of the three principal parties had clearly defined functions, responsibilities and obligations to one another at each stage of the process. This triangular relationship worked all the way down to the individuals involved in the project, the client project manager, the design project manager and the construction project manager. This is deliberate so that a spirit of co-operation is created and a common purpose in the organisation as well as promoting interaction and good communication.

2.1.4.3 Co-Location

The physical or virtual co-location of team members involved within a construction project is vital to the success of concurrent engineering as it enables all group members to contribute their views on any work being done or completed. In the case of the Glaxo project, co-location proved to be a real advantage and crucial for the projects success. It was agreed with the client before appointment that in order to achieve a successful production of such a large project was to establish the principal architect engineers as a single task force with close links at all levels with the client management team and the key representatives of the principal contractor. This design team was located in central London at Ove Arup's Howland House. This implementation of co-location was initially met with scepticism but it allowed for exceptional communication, both informal and formal. Having the majority of the design team in one building strengthened interdisciplinary collaboration and ensured that cross checking and monitoring was achieved with ease.

During the entire design process of this project constant communication was required between the principal architect engineers, client and principal contractors. It was well understood what was required of them and through the regular interaction of the parties there was very little doubt or confusion about the best route to take for the project. With all these meetings and control systems in place between all parties, the design process in relation to cost, time, and objectives ran smoothly.

When the detailed design was nearing completion, frequent meetings were held between the principal architect engineers and the principal contractor to ensure once again that everything with the designs was correct and nothing had been overlooked. Even at this stage of the design, changes could be made, but the main fundamental aspects remained the same. 'Buildability' meetings are vital to the success of any project during the construction phase, ("if you fail to plan, you plan to fail") especially one of this size. These meetings provided the tools necessary to avoid any clashes that might occur between the structural and service teams. Thanks to the excellent communication between all parties involved in the project, very little of this occurred.

It is important to note that collocation is not always feasible or possible. Where very large projects like this one occur, collocation is helpful. However for smaller projects where the individuals may be involved in various simultaneous different schemes collocation is difficult to achieve generally because of both time and cost constraints. Where this occurs concurrent engineering tools need to be used to improve the collaboration and communication between the remote participants. (See section 2.4)

2.1.4.4 Control of the Design Process

For any construction project to be successful, two basic requirements need to be satisfied. There has to be a good understanding of the customer requirements and expectations as well as constant attention to customer satisfaction. The importance of these two requirements is significant because they will result in minimum late design changes later on in the project. As a result time and money will be saved. In a concurrent engineering environment, due to the high level of interaction and communication between the different teams, continuous control of the design process is easier to achieve.

In the case of the Glaxo project, a high level of engineering was necessary due to the demanding scientific facilities required, and it was essential that a high degree of control in the design process existed. A continuous monitoring procedure enabled the Glaxo management team, the users and the facility managers to observe these designs and review them in detail. The control of this design was built around a simple framework of phases starting with the master plan through to detailed design. The level of the design was agreed with the client before the completion of each phase. The principal architect engineers then produced a set of documents at the end of each phase. The documents showed the level of design, cost and the project program. These were in turn reviewed by the scientific users and formally signed off to become the control base for the development of the designs for the subsequent phases.

2.1.4.5 Sharing Information

The sharing of information between project members is another essential aspect of a successful concurrently engineered building. Sharing information is necessary to promote the cooperation among the members of a multi-disciplinary design teams. A construction project, the size of the Glaxo Wellcome Research Centre, required an extensive amount of information to be shared throughout the building's procurement, from initial design through to the buildings completion and handover.

The principal architect engineers had the benefit of having its design team set up in a location fully linked to the main Arup data network. This allows full integration between the design teams CAD (computer aided design) and management systems. This maximised efficiency and communication during the design phase and gave access to Arup's proven systems and software. An initial CAD model of the building was created and full access was given to all the members of the design team for feedback. This allowed better coordination of the various elements that went into the design. Mechanical and electrical subcontractors could also be brought into the design task force area to extend the principal architect engineer's design into detailed installation and fabrication drawings.

The fact that the Glaxo project management team understood the importance of good communication between the different parties involved in this project meant that the

project was completed successfully with very few late design changes. There was little conflict between the various design teams and even those that did occur were resolved quickly thanks to the excellent communication and interaction.

2.1.4.6 Cost Control

Glaxo asked the design team to produce an initial estimate that was within 25% of the final out-turn cost. Some 5% of the original estimates were set aside to cover design development and a further 10% to allow for changes in the nature of the project. The combined 15% was a ceiling and any considered changes could not take the total costs above that figure. This extra money could also not be used to increase the overall size of the project. Only changes within the scope of the original concept design were permitted. The estimated cost of the project was £500 million whilst the final cost was just over this amount but well within budget.

2.1.4.7 Conclusion

The Glaxo Wellcome research centre construction project is widely regarded as a model for how the recommendations of Sir Michael Latham's 1994 report entitled *Constructing the Team* can be implemented on a large-scale construction scheme. On a project of such a large scale and complexity, control was the essence of the job. The level of control was achieved by obtaining a high level of information sharing. It was critical that all the people involved in the project were able to find out exactly what stage the project was at, at any point of time. This allowed everyone involved in the project to be in complete control of their job, since they knew exactly where they stood relevant to time, quality and completion of the whole project. This resulted in a project success. Moreover the fact that the final cost of the project was within the initial budget and completed on time indicates the level of success in terms of management and planning of the project. The degree of collocation that occurred during this project is not always possible though, especially when projects are small. The Glaxo project involved rich clients so the cost of travelling was almost irrelevant.

2.1.5 Concurrent Engineering Conclusions

In many construction projects alternative solutions for co-location and sharing information have to be used. As previously stated, co-location is sometimes not feasible, often due to the costs involved. Also when designers/engineers are working on multiple projects of smaller size they need to be based in their own offices. To allow concurrent engineering to be a success where co-location is not possible IT tools need to be incorporated. Since 1988 Information Technology and the World Wide Web have become important tools for completing many of the tasks relating to concurrent engineering. The increase in technology means that designs can be carried out quickly and with ease on desktop computers. Then, using the World Wide Web, the designers can share information over networks almost instantly so that appraisals can be carried out immediately. It has therefore greatly reduced the time spent during the design phases. Using such computer systems allows virtual co-location enabling improved collaboration and communication remotely between the different team members. In practise the use of the internet is having as desired an effect. Most companies are connected to a simple broadband connection and then sharing that connection amongst too many users making it slow. Coupled with the shear volume of data they are trying to move means that they might as well have each user connected via a modem. The following section will look at collaboration software/systems currently available.

2.2 Collaboration Systems

2.2.1 What are Collaboration Systems?

To collaborate is to work together jointly on a project and a system (OED, 2005). So a collaboration system is a system that promotes collaboration of remote users. It is usually associated with wide area and local area networks. It is the software that handles all of the communication, file sharing and various other tools that allow concurrent engineering to occur.

2.2.2 The History of Collaboration Systems

Collaborative systems are also known as Computer-Supported Cooperative Work (CSCW) the first recognised identifiable definition. The term was presented by Grief & Cashman (1984). It was not long before the IT community accepted this term and in 1986 the Microelectronics and Computer Technology Corporation (MCC) of Austin, Texas sponsored the first bi-annual international conference held in the United States, subsequently sponsored by the Association for Computing Machinery (ACM). Europe was not far behind and in 1989 the first bi-annual conference began.

Collaboration technologies have been around since the Egyptians in 3000BC. It was the Egyptians that developed papyrus, the first form of paper and arguably the first collaboration tool. They used the papyrus to write messages and notes to each other, hence the first collaboration technology (HQpapermaker, 2004). Actual paper was not invented until 105AD by the Chinese (The American Museum of Papermaking, 2004).

Perhaps more widely accepted as the original collaboration tool is the telephone, first invented on March 7th, 1876 by Alexander Graham Bell. The telephone was invented as a result of his research into improving the telegraph system. Bell was trying to improve the current system so that multiple telegraphs could be sent at the same time (his theory "harmonic telegraph" was based on the principle that several notes could be sent simultaneously along the same wire if the different signals differed in pitch). It is said that he discovered by accident that the twang of a spring could be heard over his harmonic telegraph system. Almost a year later in March 1876 Bell uttered the first

famous words into the device to his assistant in the next room "Mr. Watson, come here I want to see you". (FAQFarm, 2004)

In December 1968 Douglas Engelbart demonstrated the first networked remote collaboration with video communication and remote control. See Figure 2.3 (Christiansson, 2001)

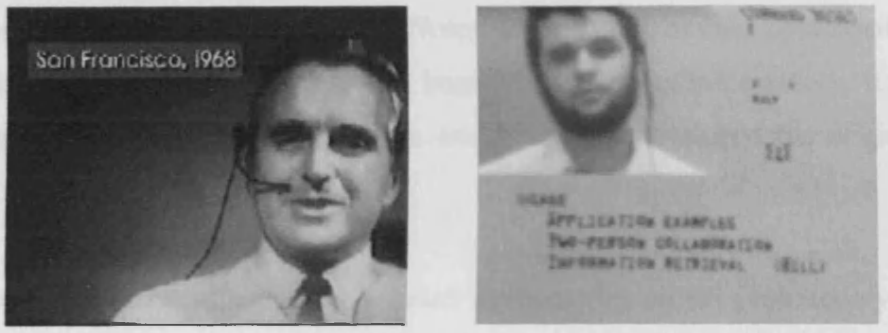

Figure 2.3 – Screen shot from the very first Video Conference

This first video conference was a public demonstration at Stanford Research Institute in Menlo Park, California and was also the first public demonstration of the mouse pointing device. The demonstration also included a shared workspace, hypertext, object addressing and dynamic file linking. Unfortunately the technology was far ahead of its time and due to the lack of bandwidth for the requited information sharing it did not become popular until the late 1990's.

The next major step in the evolution of collaboration systems was the email. Electronic mail for the internet was invented in 1971 by a computer engineer named Tomlinson (About.com, 2004). Tomlinson was part of the research team that developed the internet in 1968, however his major achievement was developing a system where by electronic mail could be sent across a network. The first email message ever sent was "QWERTYUIOP" (About.com, 2004).

The next twenty years of evolution of collaboration technology revolved around computer gaming. This is due to the profitability of computer games providing the necessary funding for the research. MUD1 (Multi User Dungeon) was the first multi-user adventure game using text based communication over a computer network. MUD1

was developed by Trubshaw & Bartle based on a popular Dungeon game of the time. (Kobb, 2001)

Lotus notes is a type of networked groupware which supports "offline collaboration" between users. The program was created in the years from 1984 to 1989 by computer programmers: Ozzie, Kawell, Halvorsen and S Beckhardt. It organises collaborators offline discussions, creating discussion threads, multiple indexing and time stamping. Lotus notes was modelled on PLATO Notes, a messaging system developed with the original computers in the 1970's to flag bugs with necessary information. It was from working with PLATO notes that Ozzie and his group developed the original Lotus Notes. (IBM.com, 2004)

Even before Lotus Notes was fully released the next step on the evolutionary ladder of collaboration technology was being taken. Oikarinen developed IRC during the summer of 1988 IRC is an acronym of Internet Relay Chat and allows real time chat between users over both local and wide area networks using a client/server protocol (Reid, 1991).

Using the previous examples of collaboration technology, the common features of computer based collaboration systems can be derived. The following list describes the most common important features of these systems:

- They run over a network (LAN/WAN);
- They support collaboration between distributed participants. Users can share information, work together on projects, ask questions, and access outside experts or trainers;
- They provide a persistent environment (archiving capability), meaning that documents are stored and available for retrieval and sharing amongst workers;
- They are normally platform independent, thus can be used on multiple kinds of computers

There are two widely accepted types of collaboration tools, asynchronous collaboration and synchronous collaboration. Asynchronous collaboration occurs when people are communicating with each other at different times, and synchronous collaboration is

when people are collaborating at the same time. As previously discussed, early collaboration tools relied heavily on asynchronous communication due mainly to the limits on technology, particularly available bandwidth. Users would rely on emails, newsgroups and forums to communicate with each other.

Chat rooms were the first synchronous communication tools to be developed. Users can log on and see all other available users to talk to via text chat. Microsoft Messenger is probably the most well known and widely used online text chat system at present. Recent surges in bandwidth availability and the reduction in cost has led to the development of more advanced synchronous based software systems. Video conferencing and interactive whiteboards have become an integral part of collaboration tools.

2.2.3 Detailed Examples of Existing Collaboration Systems

The following sections consider the more up to date collaboration tools that have emerged since the late 1980s. These systems are extremely relevant to any current research into the area, especially in relation to concurrent engineering.

2.2.3.1 COLAB

The COLAB project was born out of frustration in 1987 (Stefik et al, 1987). Researchers in PARC's (Palo Alto Research Center) Knowledge Systems Area were enthusiastic users of white boards. They used them during all meetings to scribble ideas down and help with designs. The whiteboard was an excellent tool for brainstorming, however, after meetings were finished the participants needed to copy that information into their computers. They worked with XEROX and created the first interactive whiteboard. It was a physical "liveboard" which used frosted glass and lasers for imaging, similar technology to that of a photocopier. During the early usage of these new interactive whiteboards the PARC researchers found that meetings became faster paced due to the emphasis of the communication switching from the audio channel to the video.

The problems arose due to the hardware requirements. The initial whiteboards needed quite a lot of computing power and the whiteboard itself was large and cumbersome. The PARC team envisioned portable whiteboards and "portable meetings". It was their research that led to the first software based interactive multi user whiteboard.

2.2.3.2 The MIT Dice Project

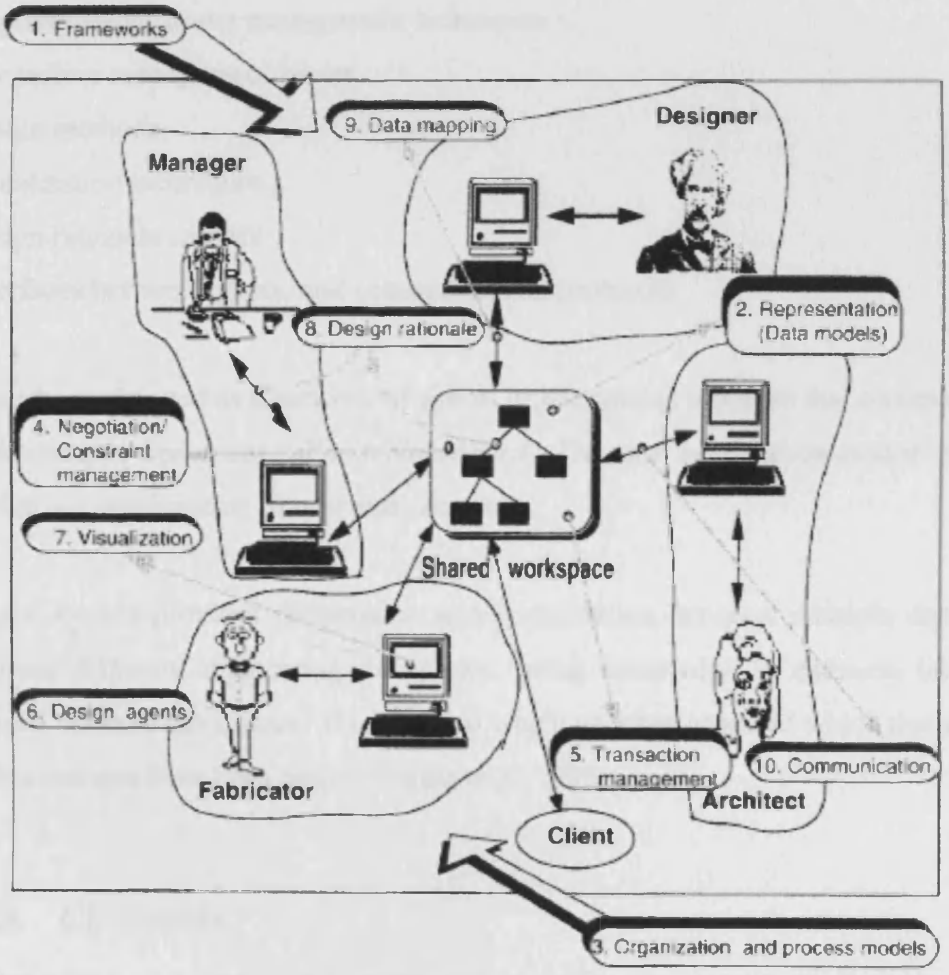

Figure 2.4 Dice Research Issues

The MIT Intelligent Engineering Systems Laboratory spent five years from 1988 to 1993 researching and developing a computer based architecture program called the Distributed and Integrated Environment for Computer Aided Design, or Dice. The program was developed for the purpose of addressing coordination and communication problems in engineering (Sriram etal, 1992).

The Dice project set a precedent for collaborative systems for engineering and is considered the benchmark for much of the research carried out in the area of collaboration for engineers. Dice addressed the following research issues, also illustrated in Figure 2.4.

- frameworks
- representation issues
- organisational issues
- negotiation/constraint management techniques
- transaction management issues
- design methods
- visualisation techniques
- design rationale records
- interfaces between agents, and communication protocols

Dice can be envisioned as a network of agents or knowledge modules that communicate through a shared workspace called a *blackboard.* The term agent when used in context with Dice is a combination of user and computer.

The Dice system provides cooperation and coordination between multiple designers often from different engineering disciplines, using knowledge to estimate interface conditions between disciplines. It can record which user has accessed which documents and what changes have been made. (Sriram et al, 1993)

2.2.3.3 CU-SeeMe

The first widely available video conferencing centre was named CU-SeeME. CU-SeeMe is a desktop video-conferencing system developed at Cornell University that was designed to accommodate multiparty video conferences over the Internet on simple desktop personal computers (Dorcey, 1995). Initially it was developed for Macintosh computers but is now available for Windows too. The software is designed to run on as many computers as possible to allow for as much interaction as possible. Because CU-SeeMe can run on low-end desktop PC machines with minimal network connectivity

and was released as freeware, it has enjoyed extremely widespread usage with over 1000 downloads per day, a huge amount for the mid 90s.

CU-SeeMe clients can be connected in a point-to-point fashion or through a central *reflector*. The reflector multiplexes multiple video streams over a single connection and gives the conference a star topology. For multiparty conferences, connecting to a reflector reduces the number of independent connections from *n2* to *2n*, where *n* is the number of clients in the conference. Over the years, the reflector has grown to include other operations, such as unwanted data pruning, bandwidth management, and transcoding. The reflector model also imposes limitations on the scalability of the system.

It is the reflectors that have proved popular with users. Reflectors set up around the world promote social interaction by providing a common place to which users may connect. Users rendezvous at well-known reflector sites to meet other people.

CU-SeeMe's transport mechanism is a best-effort protocol built on top of UDP (User Datagram Protocol). It includes a robust auxiliary transport mechanism that allows data types other than audio and video (e.g. text) to be used in a conference. The protocol provides two modes of operation for auxiliary data: best-effort streaming and reliable transport.

The CU-SeeMe codec utilizes lossless intra-frame compression on 8x8 pixel blocks of 4-bit, 160x120 greyscale video. It also uses conditional replenishment: only blocks that have changed beyond a specified threshold value are sent as part of a frame update. Standard codecs (e.g., Intel DVI) are utilized for 8 bit, 8kHz-sampled audio. (Dorcey, 1995)

2.2.3.4 CAIRO

Hussein et al (1995) worked on a project entitled CAIRO (Collaborative Agent Interaction and synchROnization system), a system for managing participants in a distributed conference. They used various existing models of group interaction and social communications theory in order to develop the CAIRO system. Unlike many

conference systems which have concentrated on the technical issues of communicating information between computers, the CAIRO team emphasised the role of the computer as the mediator and conference control mechanism. CAIRO provides both media synchronization, i.e. insuring that all information conveyed from one participant to another is synchronized, and agent synchronization, i.e. insuring effective structuring and control of a conference.

Although developed predominantly for the engineering industry, CAIRO's architecture is extensible and it can be used in many other sectors e.g. business conferences. Testing carried out at the Intelligent Engineering Systems Library showed that the CAIRO system greatly enhances the efficiency of group collaboration ensuring a satisfying and useful experience for all users.

2.2.3.5 CVW

CVW is the Collaborative Virtual Workspace and was developed in the late 90's by the Mitre Corporation. It is a prototype collaborative computing environment, designed to support geographically dispersed work teams. The software provides a virtual work space within which applications, documents and people are directly accessible in rooms, floors and buildings.

Users of the software describe CVW as a building that is divided into floors and rooms, where each room provides an area for communication and document sharing. CVW allows people to gather in rooms to talk through chat or audio/video conferencing and to share text and URLs with one another with their chat.

Document sharing also takes place within rooms. Users can share documents with anyone else in that room, allowing them to read the document or view information about the document (such as creator, description, creation date, modified date, last modified by). Document types include whiteboards, URLs, and notes etc. The rooms are always present even when there is nobody using them so the documents can remain until removed by an authorised user. This promotes persistence within the system. Figure 2.5 shows screen shots from the CVW system.

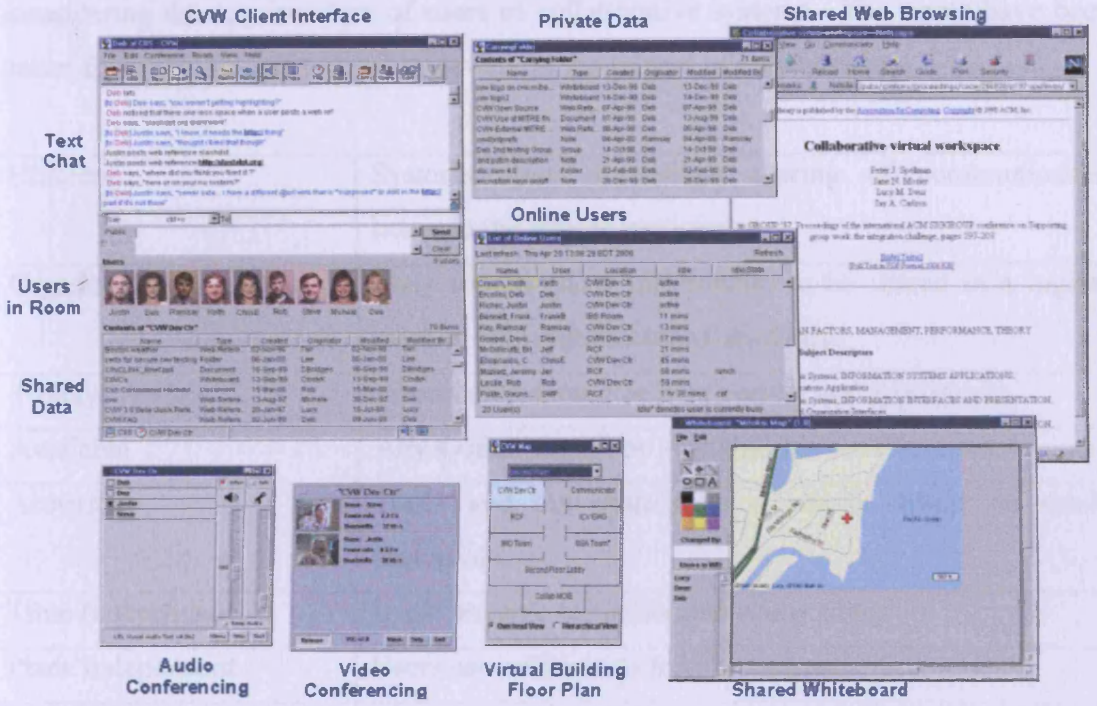

Figure 2.5 CVW

The Collaborative Virtual Workspace is also open source and has become a popular application, used a lot in industry to promote collaboration. The software promotes both audio chat and video conferencing when the bandwidth and hardware allows.

2.2.4 Collaboration Systems Conclusions

Using collaboration systems, particularly during the initial design phases of projects is becoming increasingly popular. As a result more and more collaboration systems are being developed. As the competition grows developers produce more complex systems which can handle more users and do new things.

The implications of using modern collaboration systems is that many of the recommendations set out in the Latham and Egan reports can be fulfilled and construction project costs, time and a reduction in defects can be achieved.

The following table contains various terms that seem most appropriate when considering the requirements of users of collaborative systems. The terms have been taken from a variety of sources reviewed on the subject of collaboration:

Efficient	Systems allow immediate sharing of communication between the remote participants
Organised	They should allow information to be shared in a logical manner with certain control methods.
Timely	Information should be kept current and appropriate.
Available	Any system should be available 100% of the time
Accessible	Tools and the system as a whole should be easily accessible.
Time Independent	Users are able to collaborate at any time.
Place Independent	Users can collaborate from anywhere.
Self-documenting	Tracks user communication when appropriate.
Scaleable	Enables many users to collaborate simultaneously.
Precision	Allows for precise representation of facts.
Immersive	Captures the full attention of the senses.

Table 2.1 Requirements of Collaboration Systems

These terms can act as a guide when developing a new collaboration system and learning from previous work is essential as to avoid pitfalls and ensure a successful system is developed.

CHAPTER 3

Graphical User Interface Design & Metaphors

This chapter looks at graphical user interface design and how to create a good user interface for an IT system. The first section of the chapter starts by looking at the history of the user interface so that a better understanding of them can be gained. Then what makes a good or bad interface is discussed before explaining the guidelines on creating a good user interface. The second section of the chapter looks in more detail at metaphors an important aspect of modern user interfaces and discusses how they are used when creating user interfaces. Learning the relevant skills required to create a successful user interface was paramount to achieving the projects objectives and fulfilling the projects aim.

3.1 Graphical User Interface Design

3.1.1 What Are Graphical User Interfaces?

A Graphical User Interface or GUI, often referred to as a "gooey" is the term given by computer programmers to the system of icons, taskbars, and other objects that computers use to display and access information.

3.1.2 The History of GUI's

Most of the computing world agrees that the first "real-life" usable GUI appeared in Xerox's Alto computer, developed in 1974. The Alto was seen as a smaller more portable replacement for the mainframes of its time. The Alto did not use GUI's as are used today but did use a series of graphically driven applications. It was about the size of a small car (Tuck, 2001).

There was however a much earlier reference to a GUI made by a visionary named Vennevar Bush. Bush was a scientist and futurist who worked as science advisor to President Roosevelt. He published a paper in 1945 named "As We May Think" which envisaged a computing device named "memex" which could store and create links between research documents. Bush wrote of memex:

"A memex is a device in which an individual stores all his books, records, and communications, and which is mechanised so that it may be consulted with exceeding speed and flexibility. It is an enlarged intimate supplement to his memory." (Bush, 1945)

The 'Memex' is a conceptual machine that could store vast amounts of information, in which a user had the ability to create information "trails": links of related text and illustrations. This trail could then be stored and used for future reference. Figure 3.1 shows a sketch of Bush's conceptual Memex machine.

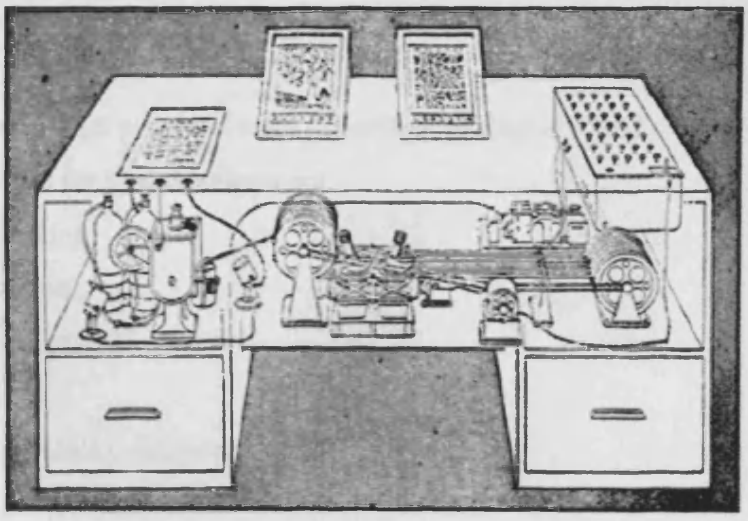

Figure 3.1 – A sketch of Bush's Memex machine

Bush's idea of memex directly influenced Ted Nelson, Douglas Engelbart, Andreis Van Dam and other pioneers of the computer hypertext. Ted Nelson tried to devise a text-handling system that would allow writers to revise, compare and undo their work easily for a term project during his Masters in humanities at Harvard. He attempted to write the project in assembler language on a mainframe, a long time before word processing had been invented and unsurprisingly his attempt fell short. It was 5 years later that he

presented his first paper at the annual conference for the Association of Computing Machinery (ACM) and coined the phrase "Hypertext". Nelson describes Bush's article "As We May Think" as describing the principles of hypertext.

Douglas Engelbart was in the navy, stationed in the Philippines in the late 1940's when he came across Bush's article "As We May Think" in a Red Cross Library. He was inspired by the paper and later on whilst working at Ames aeronautical lab he developed the idea that would form the basis of today's computer interfaces (Griffin, 1999). Engelbart began the Augmentation Research Centre (ARC) during the early 1960s, a development environment at the Stanford Research Institute. The main project of his research group was the creation of an On-Line System (NLS), the world's first implementation of what was to be called hypertext. Hypertext was just a small part of the goal of ARC. In his paper entitled "Working Together" Engelbart suggested the idea of "asynchronous collaboration among teams distributed geographically". This endeavour is part of the study of Computer Supported Co-operative Work (CSCW). "Augmentation not automation" was the slogan, the goal being the enhancement of human abilities through computer technology (Engelbart & Lehtman, 1988).

The key tools that NLS provided were (Engelbart & English 1968):

- Outline editors for idea development.
- Hypertext linking.
- Teleconferencing.
- Word processing.
- E-mail.
- User configurability and programmability.

The development of these required the creation of:

- The mouse pointing device for on-screen selection.
- A one-hand chording device for keyboard entry.
- A full windowing software environment.
- On-line help systems.
- The concept of consistency in user interfaces.

Itemising these accomplishments using today's terminology emphasises their detachment from one another. However, NLS was an *integrated* environment for natural idea processing. The emphasis was on a visual environment, a revolutionary idea at a time when most people (even programmers) had no direct contact with a computer. Input was by punched cards and output by paper tape.
Engelbart's work directly influenced the research at Xerox's PARC, which in turn was the inspiration for Apple Computers. Ted Nelson cites him as a major influence. In 1991, Engelbart and his colleagues were given the ACM Software System Award for their work on NLS.

3.1.3 Why are GUIs Hard to Design?

Most technical papers concerning the effective design of user interfaces include the statement "user interfaces are hard to design". In addition to any difficulties in actually designing the interface, user interfaces add the problems that (Myers, 1993):

- Designers often have difficulty learning the user's tasks.
- The tasks and domains are complex.
- There are many different aspects to the design which must all be balanced, such as standards, graphical design, technical writing, internationalisation, performance, multiple levels of detail, social factors, legal issues and implementation time.
- The existing theories and guidelines are not sufficient.
- Iterative design is difficult.

Myers (1993) went on to comprise another detailed list of difficulties in GUI design including "they require iterative implementation" and "there are real-time requirements for handling input events. The list itself has some relevance to collaborative systems and concurrent engineering but the majority is rather irrelevant.

The user interface of a VCR (video cassette recorder) is a good analogy of the problems faced by user interface design. Consider one of the original VCRs built in 1985 and compare that with one built today. The difference between the two models is not due purely to technological advances. Features and technological advancements aside, the progression in the understanding of the need for good user interaction and a good user

interface makes the biggest difference. Figure 3.2 shows photographs of two VCRs, one from 1985 when they first appeared and the other from far more recently.

Figure 3.2 – 1980's & Today's VCR

The model built in 1985 has an abundance of buttons readily available on the faceplate of the unit. This makes using the VCR quite daunting for many users as they are faced with a number of perhaps complex decisions. The older models also came before remote controls so there was no alternative to the user interface on the front of the system. Today's VCR only has a few buttons for the key features people use: play, fast-forward, rewind, stop, and eject. The more recent model will undoubtedly have more features than the original VCRs. However, these features are hidden behind a drop-down panel or perhaps are only accessible from the remote control, another addition to the interface of the system. This makes them accessible when needed as apposed to "in your face" which simplifies the user's interactions (Hobart, 1995).

When designing interfaces, the features used most frequently should be readily available, however it is important that the temptation to put everything on the first screen or load the toolbar with rarely used buttons is avoided.

To make the task of interface design easier programmers have devised a set of principles that should enable the efficient production of "usable" user interfaces. The following section looks at these user interface design principles in more detail.

3.1.4 Principles of Good GUI Design

3.1.4.1 The Users 'Bill of Rights'

Dr Claire-Marie Karat is a psychologist and leading computer-industry researcher in the evaluation of the way that people interact with their computers. She works at IBM designing 'human interfaces'. During her research she produced this user's bill of rights which was widely accepted by colleagues in the GUI design sector (Karat, 1998). This bill of rights can be seen in appendix A.

Although this bill of rights was produced in 1998 it is still fully relevant when considering GUI design today. The following section looks at GUI design in more detail.

3.1.5 GUI Design Basics

The following design basics are taken from the IBM website. They are based on the experiences of a large team of IBM programmers involved in the design of many user interfaces. They combine traditional wisdom with extensions to address the evolution of future interfaces such as the increasing use of 3-D and real-world representations. Thanks to the blossoming popularity of the Internet and the World Wide Web the progressions have been strongly influenced.

The IBM principles were developed during the design of an object-oriented user interface (OOUI). IBM pioneered OOUI architecture and design. Popular operating systems such as Windows 95, IBM OS/2 Warp, and CDE for Unix provide varying degrees of object-orientation for users.

Before any design principles can be used and interfaces designed the users' tasks and requirements need to be addressed. If the users requirements are not fully understood then applying any design principles is pointless as the users will not be happy with the final product.

The overall aim of a good user interface, especially for the purpose of concurrent engineering is to achieve an interface which has positive effects on the user's productivity and positively support users' endeavours without intruding adversely. It is also important that the interface is transparent to the user's task, satisfying and fun to use (IBM[2], 2004).

3.1.5.1 Simplicity: Do not compromise usability for function

It is important that the interface is kept simple and straightforward. Users benefit from function that is easily accessible and usable. An interface that is cluttered with many advanced functions distracts users from accomplishing their immediate tasks. A well-organised interface that supports the user's tasks fades into the background and allows the user to work efficiently.

Basic functions need to be obvious, while advanced functions may be less obvious to new users.

3.1.5.2 Support: Give users control and provide proactive assistance

It is important that users have control over the system, to enable them to accomplish tasks using any sequence of steps that they would naturally use. Do not limit them by artificially restricting their choices to a defined notion of the "correct" sequence.

The system should also allow users to establish and maintain a working context, or frame of reference. The current state of the system and the actions that users can perform should be obvious. Users should be able to leave their systems for a moment or a day and find the systems in the same familiar state when they return. This contextual framework contributes to their feeling of stability.

Most users perform a variety of tasks, being expert at some and novice at others. In addition to providing assistance when requested, the system should recognise and anticipate the user's goals, and offer assistance to make the task easier. Ideally, assistance should provide users with knowledge that will allow them to accomplish their tasks quickly. Intelligent assistance is like the training wheels on a bicycle i.e. at some

point, most users will want to take them off and go forward on their own. The assistance should allow them to become independent at some point when they choose to be so.

3.1.5.3 Familiarity: Build on user's prior knowledge

It is important that interfaces allow users to build on their prior knowledge, especially knowledge gained from experience in the real world. A small amount of knowledge, used consistently throughout an interface, will allow the user to accomplish a great number of tasks. Concepts and techniques are learnt once and then applied in a variety of situations. Users should not have to learn new things to perform familiar tasks. The use of concepts and techniques that users already understand from their real world experiences allows them to get started quickly and make progress immediately. These concepts and experiences are regarded as metaphors and clever, correct use of metaphors in GUI design is of paramount importance. Section 3.2 looks at metaphors in more detail as their relevance to successful user interface design can not be understated.

Many of the metaphors used in user interfaces today are inadequate when compared to the real world. Through the use of visuals and interaction techniques that more closely resemble users' real world experiences, there should be little need to continue to rely on such metaphors.

Previous user interface designs tended to invoke a principle of consistency when no single design alternative appeared to be the most suitable solution. Choosing to be consistent with something the user already understands, enables the interface to be easier to learn, more productive, and possibly even fun to use.

3.1.5.4 Obviousness: Make objects and controls visible and intuitive

It is important that real-world representations are used in the interface. Real-world representations and natural interactions (direct action) give the interface a familiar look and feel and can make it more intuitive to learn and use. Icons and windows were early attempts to draw on user experiences outside the computing domain. In an object-

oriented interface the objects and concepts presented to users parallel familiar things from the real world; for example:

- Trash can – also known as the recycle bin is where things are thrown away. An object on the desktop displayed as a trash can communicates to users that it is a place for discarding things. It looks like a trash can rather than an abstract container, and the user should be able to show its contents in a meaningful way.

- Telephone – the actions taken with telephones are so familiar to everybody that they require little thought. A telephone object on the desktop indicates to users that it will allow them to perform phone-related tasks, and users will expect it to behave like the real thing.

The controls of the system should be clearly visible and their functions identifiable. Visual representations provide cues and reminders that help users understand roles, remember relationships, and recognise what the computer is doing. For example, the numbered buttons on the telephone object indicate that they can be used to key in a telephone number.

Allow users to interact directly with objects and minimise the use of indirect techniques. Identifying an object and doing something with it (like picking up the handset of a phone to answer it) usually are not separate actions in the real world. Likewise, with direct action techniques, explicit selection is not necessary because selection is implicit in the actions users take with objects.

3.1.5.5 Satisfaction: Create a feeling of progress and achievement

It is important to allow the user to make uninterrupted progress and enjoy a sense of accomplishment. The results of actions should be reflected immediately; any delay intrudes on users' tasks and erodes confidence in the system. Immediate feedback allows users to assess whether the results were what they expected and to take alternative action immediately. For example, when a user chooses a new font, the font of all applicable text, or of sample text, should change immediately. The user can then

decide if the effect is what was desired and, if not, can change it before switching attention to something else.

Avoid situations where users may be working with information that is not up-to-date. Information should be updated immediately or refreshed as soon as possible so that users are not making incorrect decisions or assumptions. This becomes especially important in networked environments where it is more difficult to maintain state between networked systems dynamically. For example, most Web browsers display a completion percentage in the information area so that users know the progress of the graphics loading process.

3.1.5.6 Availability: Make all objects available at all times

Users should be able to use all of their objects in any sequence and at any time. Avoid the use of modes, those states of the interface in which normally available actions are no longer available, or in which an action causes different results than it normally does. Modes restrict the user's ability to interact with the system. For example, one of the most common uses of modes in menu-driven systems is the modal dialog box (such as "Print" and "Save as") used to request command parameters. Modal dialogs tend to lock users out of their system; to continue, users must complete or cancel the modal dialog. If users need to refer to something in an underlying window to complete the dialog, they must cancel the dialog, access the information they need and re-invoke the dialog.

3.1.5.7 Safety: Keep the user out of trouble

Users should be protected from making errors. The burden of keeping the user out of trouble rests on the interface designer. The interface should provide visual cues, reminders, lists of choices, and other aids. Humans are much better at recognition than recall.

Users should never have to rely on their own memory for something the system already knows, such as previous settings, file names, and other interface details. If the information is in the system in any form, the system should provide it.

During interface design the following design perspective should be adopted: users know what they want to accomplish, but sometimes they find it difficult to express their desires using the objects and actions provided, and the system is unable to recognise their request.

3.1.5.8 Versatility: Support alternative interaction techniques

A versatile interface allows users to choose the method of interaction that is most appropriate to their situation. Interfaces that are flexible in this way are able to accommodate a wide range of user skills, physical abilities, interactions, and usage environments.

Each interaction device is optimised for certain uses or users and may be more convenient in one situation than another. For example, a microphone used with voice-recognition software can be helpful for fast entry of text or in a hands-free environment. Pen input is helpful for people who sketch, and mouse input works well for precisely indicating a selection. Alternative output formats, such as computer-generated voice output for foreign language instruction, are useful for some purposes. No single method is best for every situation.

Users should be allowed to switch between methods to accomplish a single interaction. For example, allow the user to swipe-select using the mouse, then to adjust the selection using the keyboard. At the same time, users should not be required to alternate between input devices to accomplish what they perceive as a single step or a series of related steps in a task. For example, it would be tedious to require the use of a mouse for scrolling while editing text from the keyboard. Users should be able to complete an entire useful sequence through the same input device.

3.1.5.9 Personalisation: Allow users to customise

A good interface should be able to be tailored to individual users' needs and desires. No two users are exactly alike. Users have varying backgrounds, interests, motivations, levels of experience, and physical abilities. Customisation can help make an interface feel comfortable and familiar.

Personalisation of a computer interface can also lead to higher productivity and user satisfaction. For example, allowing users to change default values can save them time and hassle when accessing frequently used functions.

3.1.5.10 Affinity: Bring objects to life through good visual design

The goal of visual design in the user interface is to introduce the user in a cohesive manner to all aspects of the design principles. Visual design should support the user model and communicate the function of that model without any ambiguities. Visual design should not be the "icing on the cake" but an integral part of the design process. The final result should be an intuitive and familiar representation that is second nature to users.

The following are visual design principles that promote clarity and visual simplicity in the interface:

- *Subtractive design* - reduce clutter by eliminating any visual element that does not contribute directly to visual communication.
- *Visual hierarchy* - by understanding the importance of users' tasks, establish a hierarchy of these tasks visually. An important object can be given extra visual prominence. Relative position and contrast in colour and size can be used.
- *Affordance* - when users can easily determine the action that should be taken with an object, that object displays good affordance. Objects with good affordance usually mimic real world objects.
- *Visual scheme* - design a visual scheme that maps to the user model and lets the user customise the interface. Do not eliminate extra space in the image just to save space. Use white space to provide visual "breathing room."

All of these principles were developed by IBM and are a very general set of user interface design principles. Not all of the principles will be relevant to every user interface design but where possible the principles should be used as a check list during any interface design.

3.2 Metaphors

Metaphors are very important when considering user interface design. They are covered during the design principles, particularly in the familiarity and obviousness sections.

3.2.1 An Introduction to Metaphors

A metaphor can be defined as, "the application of a name or descriptive term or phrase to an object or action where it is not literally applicable" (OED, 1990). The word metaphor originates from a Greek word which means "carrying across". Lakoff and Johnson (1980) describe a metaphor as the way that humans understand everything in the world that is not a physical object they can directly see and understand. As soon as there is some abstraction (such as functionality, ideas, concepts) metaphorical ideals are used to help us reduce the concepts to things that are already understood.

The formulation of metaphors usually precedes the development of clear concepts. When people encounter something new which they want to learn about, they will automatically try to fit it into their existing knowledge structure, (Carroll et al, 1982). For example when faced with a new word processing package, a user will attempt to use their knowledge of a similar program and compare them. If this knowledge does not exist then the user relates the word processor to a typewriter. Therefore the use of metaphors facilitates learning and comprehension. This use of metaphors enables the user to associate everything with the real world and everyday life, something that everyone can understand without thinking.

Originally metaphors were seen to be a verbal matter, a shifting and displacement of words, whereas fundamentally it is a borrowing between and intercourse of thoughts, a transaction between contexts. Thought is metaphoric, and proceeds by comparison, and the metaphors of language derive there from. (Richards, 1936)

Richards also introduced what is now a standard terminology for the components of a metaphor:

- **Tenor**: the original concept

- **Vehicle**: the second concept 'transported' to modify or transform the tenor
- **Ground**: the set of features common to the tenor and the vehicle
- **Tension**: the effort demanded to span the gap between the tenor and the vehicle

Confirmation that a metaphor deals with thoughts rather than simply words comes from its role in the development of new concepts in science. Leatherdale (1974) and Way (1991) give some examples of the importance of metaphor to science as follows:-

The use of metaphor to extend our concepts in science is legendary. The Bohr model of the atom uses the structure of the solar system. Maxwell represents an electrical field in terms of the properties of a fluid, atoms as billiard balls, etc. This shows that even science is not the paradigm of literal language it was once considered to be, rather the metaphor is vital to the modelling processes that result in advances in science.

"New concepts are typically thought of in terms of old concepts – at least initially." The basic theory suggested by Carroll and Thomas (1982) is that when you receive new information, it goes into working memory and then you pull in related general knowledge from long term memory. These are combined to save space which is how you end up with metaphorical understandings of things. Users employ metaphors automatically when learning about something new. This is why designers of new computer systems need to anticipate and support likely metaphorical constructions to increase the ease of learning and using the system. In addition to this, guidance should be provided so that the user does not select inappropriate metaphors.

Extensive studies of metaphors and their affect on learning has been carried out by Carroll & Mack (1984). One study considered people trying to learn to use a word processor through the use of a manual, and compared the various ways they went about it. They argue that passive learning is not a great approach and that people are basically active learners preferring to use the "think aloud" method. The study suggests that users learn from doing, thinking and knowing. The research showed that users would often just try to do things without actually referring to the instructional material. Thinking meant that the Learners are often faced with situations where they perceive a need to interpret some fact or observation in order to make sense of it. In other cases,

learners are faced with the need to interpret discrepancies between what happens, and what they think should happen.

It is important to note that most users were unable to resist referring to their prior knowledge of using a typewriter as a basis for interpreting and predicting experience with word processors. This prior knowledge uses the typewriter as a metaphor for the word processing package. They also state that it is important that these metaphors do not need to be taught to the user. If this occurs, then it is contributing to the amount of material that must be learned instead of relieving the burden. The best metaphor is one that is implicitly and automatically suggested to the user merely by appearance and behaviour of a system. Such a metaphor maximises the potential savings in learning effort.

3.2.2 Metaphor Case Studies

Madsen (1994) looks at a number of case studies covering the use of metaphors in computer user interfaces. He looked at five different examples from which these three more relevant cases have been selected: design of a small command language; a design task in which users can define links between parts of different computer documents; and the design of bank automated teller machines (ATM). They are useful examples of working metaphors used in today's information technology.

3.2.2.1 A Small Command Language

An investigation of library employees revealed that the structure of their computer system was understood in terms of three different metaphors: the physical space metaphor, the conversation partner metaphor, and the organism metaphor. The investigation was based on interviews with employees who were asked to describe how they used the different computer applications at the library. The physical metaphor was identified by comments such as, "then I can go *in* and make back-up copies". This comment shows that the user is identifying the area of the computer that he/she can make back-ups as a physical space. Throughout the interviews such comments were made repeatedly. The conversation partner metaphor was identified by comments such as "well, then it *asks* me for the number of my borrowers card" and finally the identification of the organism metaphor appears in "it means that it *knows* itself what it

should do". This study led to the development of a small command language which incorporated commands such as "go in", "go to IRSystem (Information Retrieval)" and "go back". (Anderson & Madsen 1988)

This case study shows that input from the eventual users of the system is invaluable. The software developers successfully used a system of questionnaires and interviews to develop an interface that used metaphors to improve the usability of the software.

3.2.2.2 Links Between Documents

Erickson (1990) looked at a design task where users could define links between different computer documents so that where changes were made in one part of the document, the other parts were automatically updated. He wanted to use metaphors to allow the system to be used intuitively. Metaphors he considered using were the TV Broadcasting metaphor, the link metaphor and the pointer metaphor. Erickson imposed three constraints on the system: that the links are directional allowing information to be sent only from the source to the destination; that the links are one-to-many; and that the destination can not be instantly updated.

3.2.2.3 An Automated Teller Machine (ATM)

MacLean et al (1991) describes how metaphors had an important role in the design of a bank cash machine (ATM) in the United States. Originally the designer wanted to consider the ATM as an express checkout counter at a supermarket, triggering the idea that an ATM should be able to switch between an express mode with limited functions and a full version with all the services available. The system designers had personal experience of working in a bagel store (sandwich shop) in which the lengthy queues were handled by having an employee work down the queue informing the customers of the choices available and helping to fill out order forms. This meant that when the customers reached the counter their order forms could be dealt with efficiently. This familiarity lead to the design of bank cards which the customers could pre-program whilst in line.

All the case studies are an example of how metaphors can aid the development of software and products. The use of metaphors has allowed the user to develop new

cognitive structures by using metaphors to cognitive structures that have already been learnt. In the case of the ATM machine, the system designer used his own previous experience and cognitive structures to create new cognitive structures to be used by the users of his system.

3.2.3 Metaphors in Graphical User Interfaces

Metaphorical terms are especially useful in the world of information technology, particularly GUIs where there are few literal equivalents. New concepts require new terminology and it has become common practice to use metaphors rather than coin new terms. "The metaphor's role in the user interface is to facilitate learning, orientation and the forming and maintaining of the concept about the program i.e. the mental model" (Szabo, 1995).

Metaphors have become ubiquitous in the user interfaces of today's computers. They can be used in isolation for highly specific purposes or to organise and provide structure for the user interface as a whole and for the interaction between the user and the system. A pictorial representation of an airport ticket desk (as considered in section 3.2.2, figure 3.3) would be an example of a structural metaphor where files and communications functions may be accessed. A file folder is a more isolated functional metaphor. When developing a new user interface, using these metaphors is almost a necessity. If metaphors are not used, then the users may attempt to compare the system to something they already know, which may not be as comprehensive or consistent.

Typical examples of metaphorical contexts and associated familiar physical objects used to communicate the functionality and features of IT including computers, applications, electronic documents and data include these (Marcus, 1994):

- Desk: Drawers, files, folders, papers, paper clips, stick-on note sheets.
- Document: Books, chapters, bookmarks, figures; newspapers, sections, magazines, articles, newsletters, forms.
- Photography: Albums, photos, photo brackets/holders.
- Television: Programs, channels, networks, commercials, viewer guides.
- Compact disk, cassette, record, tracks, jukeboxes.
- Deck of cards: Cards, piles.

- Games, game rules, game pieces, game boards.
- Film: Rolls, slide trays, shows, reels, movies, theatres.
- Containers: Shelves, boxes, compartments.
- Tree: Roots, trunk, branches, leaves.
- Network, diagram, map: nodes, links, landmarks, regions, labels, base (background), legend.
- Cities: Regions, landmarks, pathways, buildings, rooms, windows, desks.

Typical examples of action concepts and their embodiment include these:

- Move (purposeful traversal): navigate, drive, fly.
- Browse (low goal-oriented review of options): Rapid replacement, scanning text lines, window shopping, thumbing through books.
- Scan (very rapid browsing): fast review of scrollable items, fast review of buildings, objects, people, billboards on highway at high speed.
- Locate: point, touch, encircle item(s).
- Select: touch item, poke item, grab item, lasso item, place finger on item and slide.
- Create: add (new), copy.
- Delete: throw away, destroy, lose, recycle, shred. Delete (temporary or permanent) sometimes consists of dragging a file icon to a trash can, garbage can, refuse truck, black hole, or a goat.
- Evaluate: Rotate knob, slide pointer, twist, spin.
- Pour, flow: water (pipelines, rivers), electricity.

Perhaps the most well known metaphor is the desktop which was originally developed by Xerox PARC in the 1970's and used first by Apple (Harding et al, 1997). The metaphor contains office references (desk top, documents, folders) mixed with building references (windows, trash cans). It was the PARC team that invented the GUI. They used graphics and a mouse to simplify computing, something most computer users now take for granted. In a GUI system, a mouse or joystick is used to control small graphical images of objects on the screen. The PARC GUI used a `desktop metaphor', putting icons (small pictures) of familiar objects such as folders on the screen. Instead of typing in commands, the user selected an icon with the mouse; this called up a menu (another metaphor) from which an option could be chosen.

This metaphor has been reused numerous times, most notably by Microsoft Windows. Windows 95 used the desktop metaphor to encourage more people to use computers and allow them to do so in a manner similar to the way that they would work at a desk.

Carroll et al (1988) looked at interface metaphors and user interface design. They explain that whether or not explicit metaphors are designed for a user interface, it is likely that people will generate metaphoric comparisons on their own. They found that almost 60% of errors collected from users learning word processing software were attributable to mismatched metaphor mappings. This emphasises the importance of using the correct metaphor when designing such interfaces.

Their paper suggested three stages in metaphorical reasoning: instantiation, elaboration and consolidation. Instantiation provides metaphor comparison and maintains it. It is the recognition or retrieval of something known which can be translated to the new instance. Elaboration is the more detailed analysis of the instantiated comparison. Basically it deals with the different possible ways, based on prior knowledge, that the metaphor can be used. Finally consolidation provides the control structure for the other two stages. It integrates partial mappings into a single representation of the target domain. It is a mental model. This can be interpreted as the idea that metaphors are a means to developing a mental model, rather than necessarily being the mental model themselves.

3.2.4 Choosing Interface Metaphors

Erickson (1990) describes a way of evaluating potential metaphors with regards to user interface design that has become a standard in the production of GUI's, as follows :-

- *Structure* – It is important that a metaphor has structure and there is a trade off between abstraction and too much realism. From a users point of view this means that they must have knowledge of both domains. They need to be able to map the object from the real world to the GUI.
- *Applicability* – How much of the metaphor is relevant to the problem?
- *Representability* – Is the user interface metaphor easy to represent? Distinct visual/auditory representation.

- *Suitability to audience* – Will the users grasp the metaphor?
- *Extensibility* – Does the metaphor have additional structure that may be added later? This can cause problems. If a metaphor implies too much information that is not applicable to the interface then it is a bad metaphor to use.

It is important that every aspect of the above criteria is followed when selecting a good metaphor. If a metaphor with a lot of structure is chosen then as much of that structure as possible must be used in the interface.

Integra VICE (Virtual Integrated Collaborative Environment) uses a virtual metaphor, described in detail in chapter 6. Basically the term virtual metaphor is coined because the objects are metaphors however they are actual virtual representations of the real object. The virtual metaphor used by VICE is that of a virtual multi storey office building. The fact that it uses a virtual metaphor means that the *structure,* in terms of a standard metaphor does really not apply. Erickson believed that there shouldn't be too much realism in metaphors. There would obviously be too much realism in a virtual office building as the representations are of a real office. This is why the metaphor is a virtual metaphor and slightly different to the normal metaphor. The virtual office building containing the virtual filing cabinets and many other virtual metaphors is almost fully *applicable.* The *Representability* is apparent immediately! How better to represent a working office environment than with a virtual office building. Considering the *suitability to audience* the virtual office building will be simple to use and the user will be able to fully grasp the interface as a whole. Users will interact within the interface in the same way that they would go about interacting in a normal office building. The final criterion for selecting a good metaphor is *extensibility.* The extensibility within an office building is easy to achieve. To add new applications to the system, you just need to develop another virtual metaphor within the all encompassing virtual metaphor of the office building.

Carroll et al (1988) come up with a similar method of designing with metaphors. They propose four steps: identifying metaphors; identifying metaphor/software matches; identify mismatches and implications; manage mismatches.

Combining the ideas in the two papers leads to a more comprehensive step by step method of choosing a suitable metaphor: The first step is to identify all the metaphors that can be considered for the interface. The next step is to identify those which are best suited to the interface. Then the metaphor that is both best for the interface and easiest to represent is chosen. The designer must then look for mismatches that may occur when the user is mapping the metaphor into his/her knowledge base. These mismatches are then minimised so that they do not interfere with the user's interaction with the system. This may be through informative messages, documentation tool for training etc.

3.2.5 Opposition to the use of Interface Metaphors

Constantine (1998) carried out a study of the popularity of using metaphors in user interface design. He claims that the great success of a handful of simple metaphors such as folders and trash cans eclipses the fact that the majority of metaphors fail to improve usability and many make matters worse.

Metaphors may be both literal and explicit, as in the case of the pictorial representation of an office or they can be implicit as conceptual shapers of design without a direct representation on the visual interface. For example in the case of the desktop metaphor, no actual desktop is literally represented on the screen, although the concept of the desktop is used as an underlying theme for organising the on-screen presentation.

Problems in usability originate from the misuse of metaphors which can come in various forms. Some of the biggest errors occur when real-world objects and their behaviours are simulated on-screen, whether as structural or functional metaphors. Objects which may be simple to use in the real world can become complex and cumbersome to use within an interface.

The situation for the user is even worse when real-world metaphors are employed in ways that violate the user's expectations of that metaphor. When the cognitive structure is employed by the user but does not function as it is expected to it leads to confusion. Mixing metaphors is another way of distracting the user. Most waste baskets do not burst into flames when something is thrown into them.

Figure 3.3 – User Interface for Major Internal Airline of USA until 1998

Figure 3.3 is an example of a web based user interface that uses a structural metaphor inappropriately. The interface was the main home page for a major internal airline of the USA until 1998. It is taken from Constantine (1998) and formed part of his study of the misuse of metaphors.

The usability problems caused by the use of this "ticket counter" metaphor, where users are supposed to be able to reach various facilities on the site are abundant. An obvious example of the ineffectiveness of the interface is the fact that a large arrow saying "try me" is needed to draw the user's attention to the reservation telephone.

Online reservations have become the major revenue maker for these airlines, especially with the popularity of e-tickets and yet here you have to hunt for the link. Even checking schedules and frequent flyer points is made more difficult by the sidewise labelling employed and obscured by a simulated box.

One of the strongest attacks on interface metaphors comes from Nelson (1990) who identifies three 'elements of bad design', one of which he termed 'metaphorics'. "I would like to venture that this 'metaphor' business has gone too far. Slogans and catchphrases are all very well, and these things have their uses for people who are going to learn software *approximating* rather than *understanding*." This is followed by claims that, "the metaphor becomes a dead weight," and suggestions that the "alternative to

metaphorics is the construction of well-thought-out unifying ideas, embodied in richer graphic expressions that are not chained to silly comparisons."

Although metaphor might be a 'dead weight', there is little or no evidence from Nelson of a successor that will provide a metaphor-free alternative. His alternative was VisiCalc and using hypertext was a future alternative. VisiCalc may have been a new concept for most of its users but it is still a metaphor. The program is obviously based on spreadsheets of a type already in manual use by accountants.

Hypertext was initially developed from a book or document metaphor, with links taking the user from one page to another. Apple recognised the limitations of the desktop metaphor when dealing with hypertext and used a 'card index' metaphor for HyperCard. This conflicted with their existing interface guidelines based on the desktop (Apple 1987). They corrected this by providing a new set of guidelines for HyperCard (Apple 1989). The hypertext principle has now expanded to hypermedia and hyperspace. Some have expanded the book metaphor to cope with these more expansive demands (Rauch 1997) or extended it to libraries (Pejtersen & Goodstein, 1988). Hypertext has undoubtedly become a very useful tool, but many developers find it is only part of the answer and are searching for suitable metaphors to help prevent users becoming 'lost in hyperspace'.

Kay (1990) also criticises the over use of metaphors, "One of the most compelling snares is the use of the term metaphor to describe a correspondence between what the users see on the screen and how they should think about what they are manipulating. My main complaint is that metaphor is a poor metaphor for what needs to be done. At PARC we coined the phrase user illusion for what needs to be done." He continues, "it is the magic... that really counts" and calls for a greater use of 'magic'. Magic can be explicit, like the use of teleport devices in games. More often however, the examples are more mundane, such as the ability to paste an unlimited number of times when using the 'cut and paste' metaphor. This is referred to as 'magic'.

Brown (1995) suggests that the desktop metaphor should be considered a global metaphor encompassing the whole application. This is due to the metaphor being made up of a collection of other metaphors which perhaps would not be associated with a

desktop. As a result the desktop metaphor is considered a quagmire because reality has diverged from the metaphor. Consider the desktop and how it deviates from reality. The trash can or recycle bin is a wonderful metaphor for the delete function; however trash cans are generally not situated on top of a desk. In reality people do not have icons on the desktop but use real items such as files and sheets of paper. The vertical aspect of the desktop also subverts the metaphor. It's closer to a refrigerator with randomly placed different magnets.

The global metaphor is an example of the "bigger is better" mentality. Due to the wide use of metaphors in information technology, people are now assuming that the more all encompassing a metaphor, the more useful it is. However, the usefulness of a global metaphor is actually governed by the overall goals of the interface itself. Some goals are not best suited to a global metaphor. If the aim of the interface is to input large quantities of data quickly and effectively, a global interface would be more of a hindrance than it is useful.

Various metaphors have been proposed for collaborative work spaces; perhaps the most common is the room metaphor. A number of researchers have independently explored the use of the room metaphor. Xerox PARC (Henderson 1986) developed a room concept to be used by one user at a time on a single machine, while the concept was extended to multi-user groupware by Bellcore (Root, 1988) and Condon (1990). Condon also explored the combination of the room metaphor for informal, real-time work with a form-based metaphor for formal, non-real-time work (Hämmäinen, 1991). Other researchers have gone beyond the immediate room to include balconies, doors and corridors (Pemberton, 1993).

The combination of hypertext and multimedia on the Internet has led to a series of communications or link-based metaphors, such as the World Wide Web, the Information Superhighway, or simply, the Net. Many of the suggested interfaces for future systems are based on VR and a number of metaphors have been suggested for managing these virtual spaces. Many are based on extended spaces and landscapes or on various types of community. These include fields, villages, rivers and highways (Florin 1990), farms, including information fields (Bernstein, 1993), information forests (Rifas, 1994), or urban metaphors such as the city (Dieberger, 1994).

3.3 Conclusions

To create a successful graphical interface using metaphors there are key things that need to be understood. The interface should not be too complicated. It has to be usable both by experience computer users and novices. All of the tools that are associated with the interface need to be obviously apparent. The use of metaphors will achieve this but there are other risks involved with choosing and using metaphors. The metaphor has to be relevant and applicable to the prior knowledge that it is trying to link to. As long as careful consideration and planning is used when choosing metaphors for GUIs the interface should be a success.

CHAPTER 4

Virtual Reality & VRML

This chapter shows the research that ensured the more specific knowledge and programming skills were gained to write the user interface of choice. The chapter starts by explaining virtual reality before examining a variety of virtual environments, both immersive and non-immersive as well as distributed environments. The second part of this chapter looks at VRML, the programming language chosen to create the virtual worlds. It describes the specification of the language and looks at how the language is written and compiled.

4.1 Virtual Reality (VR)

4.1.1 What is Virtual Reality?

The Oxford English Dictionary defines virtual reality as the computer-generated simulation of a three-dimensional image or environment that can be interacted with by using special electronic equipment (Oxford University Press, 1994).

A more detailed definition is states that virtual reality is effect created by generating an environment that does not exist in the real world. Virtual Reality refers to a suite of technologies supporting intuitive, real-time interaction with three-dimensional computerised databases. Virtual environment and virtual world are synonyms for virtual reality. "Immersive" VR, whereby users don a head-mounted display and interact with a 3D world using special hand controllers or gloves, is one variation of VR interface technology. More popular are interfaces based on standard desktop screens or large data/video projection displays (in 2D or 3D) used in conjunction with ergonomically acceptable desktop controls (MOD, 2004).

The rationale for virtual reality systems is that they allow users to complete many tasks almost exactly as they would in reality. This has relevance to all sorts of applications and has been used widely in gaming. A good example of clever use of VR is in safety systems where large scale high risk scenarios can be tested with users in a virtual environment which is entirely safe.

There are 3 broad styles of interaction or techniques in VR:

- **Constrained Path VR** is a term applied to a human system interaction style whereby limited or fixed motion paths or "corridors" have been defined within the virtual environment. For example, users may be free to move forward or backwards through the three dimensional scene and, at any point, stop and "look around", using whatever human interface technology has been considered appropriate for such actions.

Figure 4.1 – Immersed User Wearing Modern Headset and VR Glove

- **Panoramic VR** is a special case of "Constrained Path VR" in that digital photographs are used to create a 360° panorama or "vista" of a particular environment. The user can explore the environment by "jumping" between predefined points or "nodes" in the environment. Each node is associated with a

high quality panorama which gives the user the opportunity to look around, up and down using purely mouse controls. Areas within panoramas can be endowed with interactive "hot spots" which can then be linked to features such as databases, simplified 3D objects, other panoramas, even IETMs (Interactive Electronic Technical Manual).

- **Free Play VR**, in contrast to "Constrained Path VR", is a term applied to a human system interaction style whereby the user is free to explore the virtual world and interact with whatever component he or she is interested in. In an extreme (and often unsatisfactory) case this will allow the user a full 6 degrees of freedom motion in the virtual world (translation in x, y and z, plus roll, pitch and yaw). It is important that the computer system is able to record the movements of the user's virtual representation constantly during Free Play VR. Otherwise it will not be able to calculate intentional and unintentional collisions with features of the virtual environment. It is this reason that requires a far greater computational overhead when allowing Free Play VR. (MOD, 2004)

4.1.2 The Origins of Virtual Reality

Autodesk combined with computer company VPL on June 6th 1989 and announced their new technology "virtual reality", (VR) day at two trade shows. The announcement was preceded by four months of media coverage and hype (Bricken, 1990).

However, the idea of virtual reality was introduced to world far earlier than this through the median of science fiction. "Jacking in" to a dataspace originated in William Gibson's 1984 novel Nueromancer. This novel coined the term "cyberspace": the "consensual hallucination" of high-definition immersive graphical representation of data. The idea of virtual reality was predated by some of the technology that has become associated with it. The head mounted displays used in today's immersive systems were built by Ian Sutherland in the late 1960s and developed further within military and aerospace applications (Chesher, 1994). Figure 4.1 shows a user wearing a head mounted display and glove.

Before the commercial release of virtual reality by Autodesk and VPL, virtual reality systems had been developed by the military and NASA. Thomas Furness II developed the American Air Forces "super cockpit" virtual environment systems. These initial systems had very limited publicity and were very specialised and expensive. NASA's VIVED (Virtual Visual Environment Display system) project was developed by Michael McGreevy in 1985. VIVED was the first low-cost, wide field of view, stereo, head-tracked, head-mounted display (McGreevy, 1993).

4.1.3 Immersive Virtual Reality

4.1.3.1 The CAVE

The CAVE (Cave Advanced Virtual Environment) is a projection based virtual reality system that was developed at the Electronic Visualisation Lab, part of the University of Illinois, Chicago. It was predominantly the project of Carolina Cruz-Neira, Dan Sandin, and Tom DeFanti and was completed in the early 1990's. The CAVE was designed to be a useful tool for scientific visualisation and was presented at the SIGGRAPH 92 showcase. As stated above virtual reality is best defined as the wide-field presentation of computer generated, multi-sensory information which tracks a user in real time.

The CAVE is a multi-person, room-sized, high-resolution, 3D video and audio environment. Graphics are rear projected in stereo onto three walls and the floor. These graphics are then viewed using stereo glasses. The user wears a position sensor whilst moving within the display boundaries. This sensor is linked to a supercomputer which updates the correct perspective and stereo projections of the environment. The images are then able to move with and surround the viewer. This enables the stereo projections to create 3D images that appear to have both a presence inside the projection room and outside continuously. To the viewer wearing the stereo glasses the projection screens become transparent and the 3D image space appears to extend to infinity. For example, a tile pattern could be projected onto the floor and walls so that the viewer sees a continuous floor which appears to extend far beyond the boundaries of the projection room. 3D objects such as tables and chairs would appear to be present both inside and outside this projection room.

As far as the viewer is concerned these objects are really there. It is only when the user attempts to physically touch the objects or walk beyond the boundaries of the projection room that they will find otherwise. Many rips and tears on projections screens have occurred where viewers have forgotten to be careful when walking within these invisible boundaries. Figure 4.2 is a graphic showing a user within the CAVE (Cruz-Neira et al, 1992).

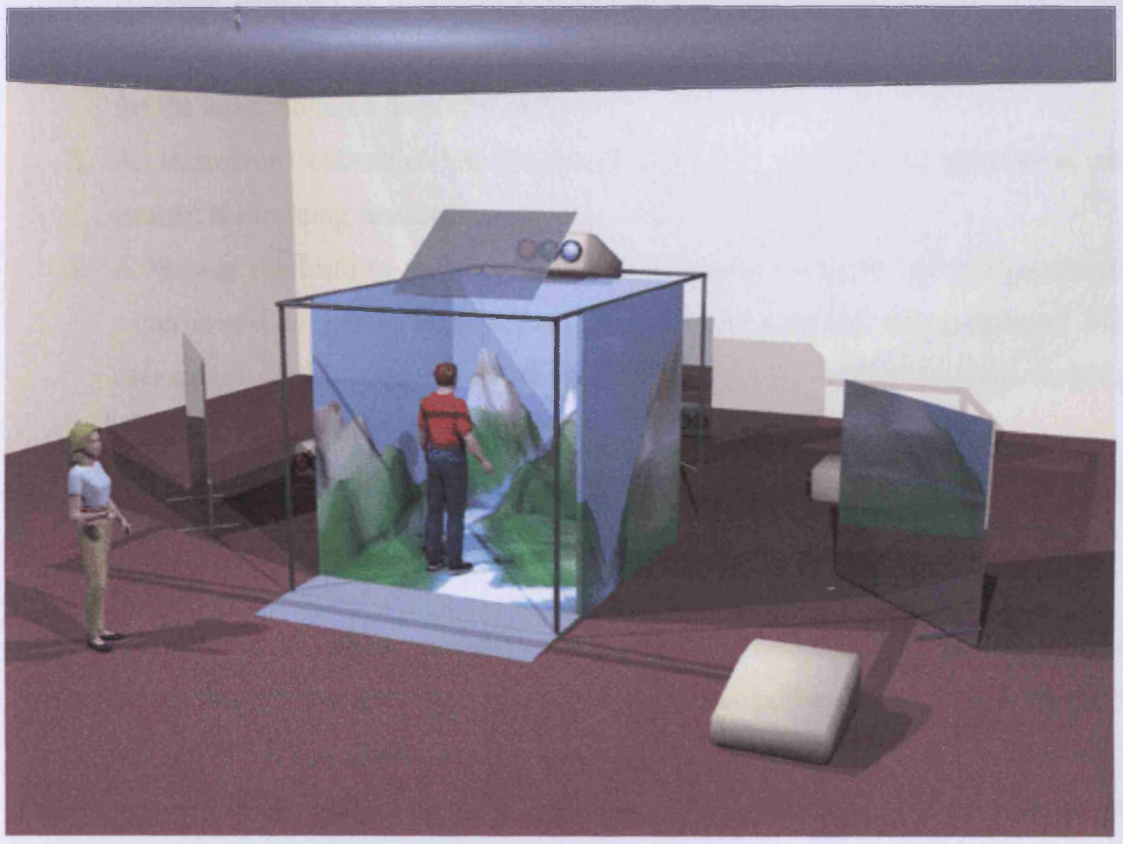

Figure 4.2 – The CAVE

The CAVE is a 10x10x9 feet theatre, made up of three rear projection screens which handle the front, right and left walls and a down projection screen which is responsible for the floor. Electrohome Marquis 8000 projectors throw full colour workstation fields (1024x768 stereo) at 96 Hz onto the screens, giving approximately 2,000 linear pixel resolution to the surrounding composite image. Computer controlled audio provides a sonification capability to multiple speakers. The user's head and hand are tracked using Ascension tethered electro magnetic sensors. Stereographics' LCD stereo shutter glasses are used to separate the alternate fields going to the eyes. A Silicon Graphics Power Onyx super computer with three Infinite Reality Engines is used to create the imagery that is projected onto the walls and floor (Cruz-Neira et al, 1993).

4.1.3.2 SAVE

SAVE (Safety Virtual Environment) is a safety training system using virtual reality which enable users to be placed into unsafe situations. For example, within an oil refinery to monitor how they deal with the refineries challenges. SAVE comprises four major parts or modules:

1. A visual Simulation which represents the core part of the system where the simulation is computed, all user input is processed and the images are generated for the head mounted display (HMD).
2. An Instructor Desk which lets the human instructor supervise the simulation and control the training session.
3. A Motion Platform to enhance the immersive experience by providing motion patterns and automatic slope adjustments according to the virtual ground. The user stands on this platform and can feel vibrations near virtual engines, shaking ground or shocks from explosions.
4. A Desktop Authoring Tool to construct new scenarios or manipulate existing ones.

The tool supports all tasks necessary to build a scenario, including the virtual environment with a 3D editor, dependencies and actions in the event network through visual programming, the graphical user interface (GUI) of the instructor with a GUI builder, as well as all motion patterns, sounds and other special components (Haller et al, 1999).

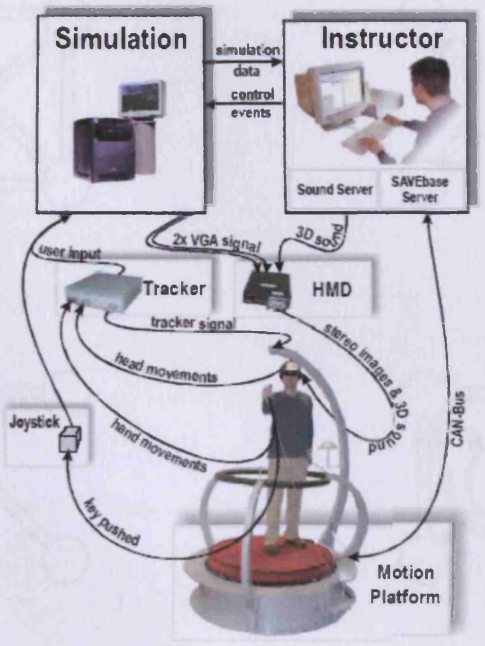

Figure 4.3 – Topology of SAVE

Figure 4.3 shows the topology of the SAVE system. A tracking system provides position and orientation of the trainee's head and hand to make sure that the graphics are rendered in the correct way. There is one tracker sensor mounted on the HDM (Head Mounted Display) to get the user's head position. The second unit is incorporated into the two button joystick used for navigating and interacting within the virtual environment. Collision detection between the trainee and the objects in the virtual environment contributes to the impression of being part of the simulation.

4.1.4 Distributed Virtual Reality

Distributed virtual reality occurs where simulated worlds run on more than one computer system simultaneously. The computers are connected over a network or even the internet. Users are able to connect in real time, sharing the same virtual world.

There are two different network topologies which are generally used for distributed virtual environments, either peer to peer or client-server protocols.

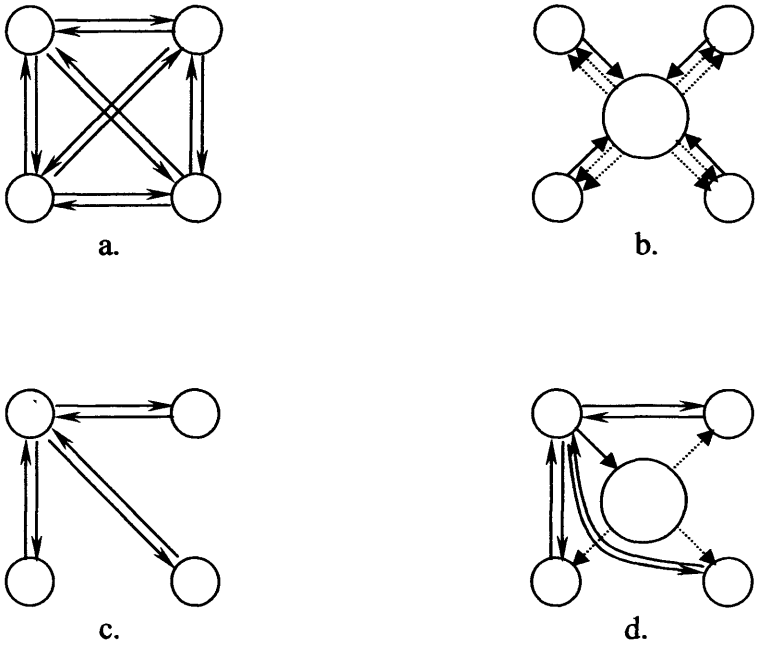

Figure 4.4 – Networking Topologies

Figure 4.4 shows the four different networking topologies generally used when developing distributed virtual environments. Figure 4.4(a) shows a peer-to-peer topology with unicast network. Fig. 4.4(b) is a peer-to-peer topology with multicast network, 4.4(c) is the client-server topology with unicast network and finally figure 4.4(d) is the client-server topology with multicast network.

Unicast allows the sending of messages to each other entity on the network for distribution. Multicast means that a subset of workstations can communicate with each other using connectionless messages. The underlying network should support the multicast operation.

4.1.4.1 Peer-to-Peer

In peer-to-peer design, the system is arranged with a set of workstations that communicate with each other over a network, where each host can send messages directly to any other host. No single dedicated host is responsible for serving other hosts requirements. Peer-to-peer topology can be implemented using unicast messages, by connecting each node to every other. This means that when a user changes the state of an entity (e.g. moves an object, even himself) a message is sent to every other participant. If a network supporting multicast messages is available, then hosts can send their message to a subset of hosts at once. This decreases the complexity of distributing

single state update messages, therefore speeding up the communications within the environment.

4.1.4.2 Client-Server

The Client-Server network protocol promotes the communication between client computers managed by server hosts. Clients do not send messages directly to other clients, but instead send them to servers which forward them to other clients and servers participating in the same distributed simulation.

The use of servers provides additional advantages such as processing messages before propagating them to other clients. They can even remove, augment, or alter the messages before sending them on to the remaining clients. Consequently, the server may determine that a particular update message is only relevant to a small subset of clients and then propagate that message only to those clients. The processing technique will depend on the nature of interaction among the entities in the virtual world. Message processing could also be achieved in peer-to-peer systems at each host, However, mapping interactions to multicast addresses or lists of receiving hosts afterwards is not easy and may be CPU-intensive. The client-server protocol easily solves this problem by moving the processing load of messages from each host to high-end server, leaving host resources for other networked virtual environment tasks.

In this topology, clients generally connect to the servers through a bidirectional unicast link. Similar to the peer-to-peer systems, the distribution of the state updates from the server to clients can be done using unicast or multicast messages.

4.1.4.3 Hybrid Topologies

Both peer-to-peer and client-server protocols can be combined to create hybrid topologies. This hybrid is then able to utilise the advantages of both approaches. For example a topology where clients connect to servers using unicast networks, however the servers pass the data to other servers through multicast messaging. Figure 4.5 shows a hybrid network where the client-server protocols are linked to the multicast peer-to-peer protocol.

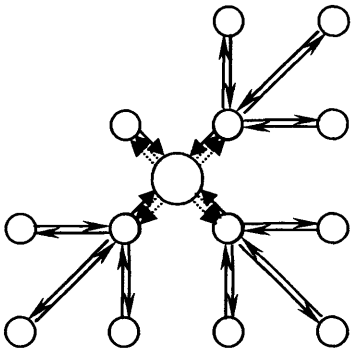

Figure 4.5 – Hybrid Topologies

4.1.5 Examples of Distributed Virtual Reality

Most virtual environments can be both non-immersive and immersive. The following examples are predominantly used without user immersion, using desktop computers with suitable input devices. To use the environment "immersively" a simple plug-in is required and then the necessary hardware. This will include hardware such as the HDM and glove shown in figure 4.1 above.

4.1.5.1 DIVE

The Distributed Interactive Virtual Environment (DIVE) was developed at the Swedish institute of computer science. DIVE is an experimental platform for the development of virtual environments, user interfaces and applications based on shared three dimensional synthetic environments. DIVE is specifically tuned to multi-user applications, where several networked participants interact over a network (Carlsson & Hagsand, 1993). DIVE is one of the most well known distributed virtual environments available on the web and was also one of the first to be developed.

Figure 4.6 is taken from the DIVE homepage (DIVE, 2004) and shows a selection of virtual environments that have utilised the DIVE system. DIVE is a peer-to-peer approach with no centralised server, where peers communicate by reliable and non-reliable multicast, based on an IP multicast. Conceptually, the shared state can be seen as a memory shared over a network where a set of processes interacts by making concurrent accesses to the memory.

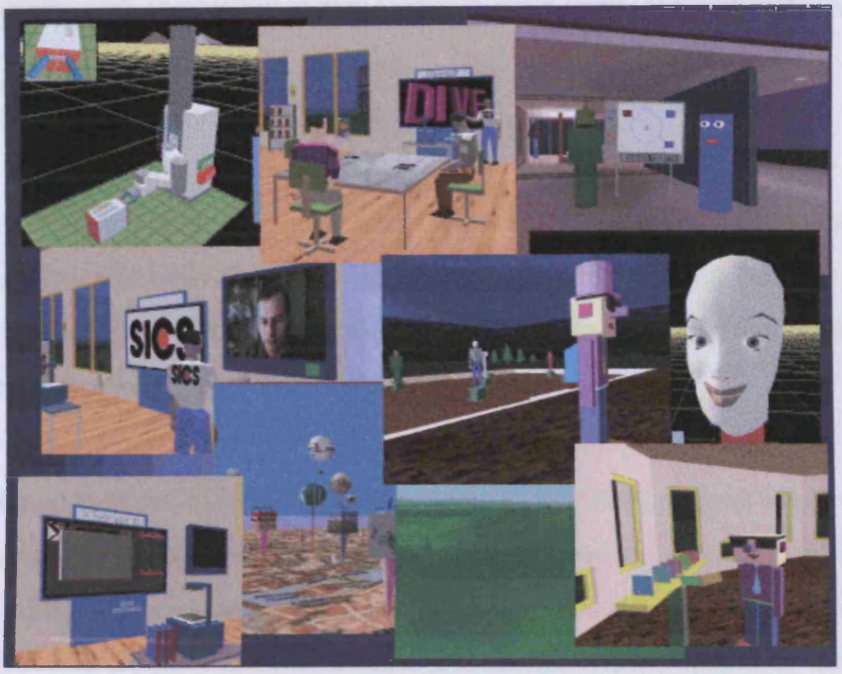

Figure 4.6 – A Selection of DIVE Worlds

Consistency and concurrency control of common data (objects) is achieved by active replication and reliable multicast protocols. Objects are replicated at several nodes where the replica is kept consistent by being continuously updated. Update messages are sent using multicast so that all nodes perform the same sequence of updates.

The peer-to-peer approach without a centralised server means that as long as any peer is active within a world, the world along with its objects remains "alive". Since objects are fully replicated (not approximated) at other nodes, they are independent of any single process and can exist independently of their creator.

Users navigate the 3D space meeting and collaborating with other users and applications within the environment. A participant in a DIVE world is called an actor, and is either a human user or an automated application process. An actor is represented by a virtual person (or avatar), to facilitate the recognition and awareness of ongoing activities.

An avatar is the virtual representation of the user within the virtual world. It is placed at the viewpoint of the user, i.e. the position within the virtual world from which the user looks at the scene. When used in a single-user world the avatar is only used to

detect collisions with objects but in the case of a multi-user virtual world, the avatar is also the visual representation of the user. The term avatar comes from an Indian religion meaning *god's incarnation on earth* (Diehl, 2001).

In a typical DIVE world, a number of actors leave and enter worlds dynamically. Such applications typically build their user interfaces by creating and introducing necessary graphical objects. Thereafter, they "listen" to events in the world, so that when an event occurs, the application reacts according to the control logic.

The Distributed Interactive Virtual Environment (DIVE) is an internet-based multi-user VR system where participants navigate in 3D space, seeing, meeting and interacting with other users and applications. DIVE software is a research prototype covered by licenses. Binaries for non-commercial use, however, are freely available for a number of platforms (Carlsson & Hagsand, 1993).
The advantages of DIVE are:

- It provides a tool for the general, portable, development of applications.
- VE tasks can be decoupled from the application through the Tcl/DIVE toolkit and its set of application programs.

The disadvantages are:

- DIVE requires a good knowledge of the toolkit to programme any new behaviours for new applications.
- It does not provide efficient embodiment representation and communication mechanisms between participants hosts.

4.1.5.2 MASSIVE

Model, Architecture and System for Spatial Interaction in Virtual Environments (MASSIVE) is another example of a distributed virtual reality system. It is an experimental distributed virtual reality system which was developed to support collaborative activity. The particular emphasis of the MASSIVE project is a large scale multi-user environment that can handle hundreds if not thousands of different simultaneous users. MASSIVE aimed to provide rich forms of interaction which draw

on real-world behaviour to make them useful and controllable in highly populated virtual worlds.

The main difference between MASSIVE and distributed VR systems that are based on a shared database approach such as DIVE (as discussed in the previous section), is that the distributed model used in MASSIVE is an independent computational processes communicating over typed peer-to-peer connections (using standard internet transport protocols). Each computational process can have multiple interfaces through which it interacts with the rest of the system. Each interface is characterised by a combination of remote procedure calls (RPCs), attributes and streams (Greenhalgh & Benford, 1995).

The advantages of MASSIVE are:

- Many participants using various types of equipment (from high-end graphics to text-based interfaces) can be present within the same environment and are able to communicate with each other.
- The spatial model of interaction allows the system to map the real world interactions to the interactions between participants.

The disadvantages are:

- Any new simulations or environments cannot be developed rapidly.
- The authors do not present details on participant embodiment.

4.1.5.3 dVS

dVS (Grimsdale, 1991) is a commercial virtual reality system which is sold by Division Ltd. dVS provides an immersive visual and auditory virtual environment software system capable of supporting multiple users easily, due to its distributed architecture. The system aims to provide a modular line for creating and interacting with virtual prototypes of CAD products. The architecture of the system is based on dividing the environment into a number of autonomous entities, and processing them in parallel.

dVS is based upon the VL (virtual library) distributed database, which is an object-orientated virtual environment control interface. Each distributed database runs across a file configured set of nodes and is managed by a single agent process on each node. All

of the various communication and information sharing that takes place within the virtual world is via this database.

The database is divided into disjointed named environments which other processes may connect to. Processes can create and destroy instances in the database, representing the state of the virtual world, and can request asynchronous notification of the creation, deletion and updating of database items. User events are realised by creating special instances in the database and using the update notifications from these instances as events in their own right. Database items and events are distributed on request (Thalmann, 1995).

The advantages of dVS are:

- Usability – a CAD designer can easily utilise the system and its features by using the supplied functionalities.
- Portability – it can be used on SGI (Silicon Graphics Incorporated) workstations which are designed specifically for 3D design and PCs.
- Efficiency – the elements can be optimised for the underlying computer system. For example the system uses the IRIS Performer on SGI Workstations.

The disadvantages are:

- dVS is not designed for multi-user applications with a dynamic number of participants and integration of different applications.
- It is not possible for two application developers to connect their animation programs within the same world.
- No mechanisms exist to enable participants to distribute their avatars to remote participants: avatar files have to be uploaded to remote hosts by using ftp protocol explicitly before connecting to virtual world.
- The avatar configuration file is too limiting for animating body gestures.

4.1.5.4 MR Toolkit

The MR Toolkit (Shaw et al, 1993) has been developed at the University of Alberta and is a toolkit for creating virtual reality style user interfaces. The original system development focussed on single-user systems, but the peer package provides some

support for multi-user systems (Shaw & Green, 1993). Communication between processes is via shared data structures, the values of which may be copied between cooperating processes. A typical single user MR Toolkit application comprises a programmer defined collection of cooperating processes, the exact organisation of which is defined by the programmer. These applications are fairly static in their configuration.

The peer package handles the multi user requirements. It allows peer-to-peer communication between such applications over the user datagram protocol (unreliable message passing). One process on a named machine and specified port will define the world. Other processes will connect to this machine and will then have to port to join the world. The communications support is low level, and the application programmer would have to implement a dynamic distributed database over the message passing provided by the Toolkit. This provides scope for flexible implementations, but leaves a significant amount of work to be done by the system user. If a type negotiation layer were built into the database functionality then the database could be made run-time extensible.

The advantages of the MR Toolkit are:

- The VE is divided into four components: presentation, interaction, geometric model, and computation. This allows the multi user access as these components can be distributed among the nodes in a network.

The disadvantages are:

- There is a distinct lack of available communication tools available and to incorporate communication between the programmers a lot of extra work needs to be carried out.

4.1.5.5 SPLINE

Scaleable Platform for Interactive Environments (SPLINE) was developed by the Mitsubishi Electronic Research Lab (MERL). The project was led by Walters and Anderson and its objective was to create open interfaces that facilitate interoperability between virtual environments built by different users. SPLINE includes both open

interface definitions for the network programmer as well as the application programmer (Waters et al, 1997).

The SPLINE system is very similar to DIVE. It uses peer-to-peer communication and a derivative of SRM (Scaleable Reliable Multicast) to reduce the number of messages passing, which in turn reduces the network load and increases the overall scalability. This method uses multicasting heavily, makes communication entity based and bases reliability on a negative acknowledgement request/response scheme.

Like DIVE, SPLINE has evolved from a pure multicast approach to a mixed client-server and multicast approach, to enable it to cope with the low bandwidth users. SPLINE divides the VE into sub-regions called locales, each associated with a multicast group.

The SPLINE Diamond Park Application is a virtual park which consists of a square mile of detailed terrain. The terrain is capable of visual, audio, and physical interaction. Participants navigate around the scene by cycling and can use an exercise bike as the physical input for this cycling. The avatar then moves on a virtual bicycle around the virtual environment. The speed of the virtual bicycle is calculated from the force applied by the exercise bike user.

The advantages of SPLINE are:

- The system has proved to be effective on pilot applications.
- The Diamond Park application could be developed using the system in a short time and integrated without major problems.
- Using the exercise bicycle interface and mapping it to virtual bicycles increases the relationship between the user and the virtual environment, in particular the embodiment through the avatar.

The disadvantages are:

- The embodiment of participants has very simple behaviours because cycling is the users' only option.
- Participants only navigate and communicate with each other using audio.
- Overall interaction within the environment is minimal.

4.1.5.6 NPSNET

NPSNET (Zyda et al, 1992), developed at the Naval Postgraduate School, is a networked VR system designed for military training and simulation with large numbers of participants. It has been called a low-cost version of SIMNET, which was the first and most important military virtual environment system and stood for Simulator Networking. The system is based wholly on standard Silicon Graphics workstations connected by ethernet.

NSPNET uses the standard DIS (Distributed Interactive Simulation) protocol which uses position and speed information to make dead reckoning extrapolations of the positions of remote objects. This means that the next position of any entity is computed based on their last received position, velocity and acceleration. Updates are multicast to all participants in the world. The system is geared towards military exercises, and objects are constant in their appearance and generally quite dispersed. The system is able to take advantage of this dispersal limiting network updates, allowing very large numbers of users to share a single world.

The advantages of NSPNET are:

- NSPNET succeeds in providing an efficient large-scale networked VE using general purpose networks and computers, and using a standard communication protocol, DIS.
- Using a multithreaded approach facilitates efficient computation over multi-process architectures.
- For his role in the VE the user can select a set of input techniques for interaction.

The disadvantages are:

- NSPNET lacks properties of generality (battlefield simulation), modularity, portability (works predefined hardware), and rapid development of new applications.
- DIS traffic handling at application level creates complexity, thus demanding more computational power.

4.1.5.7 VLNET

Virtual Life Network (VLNET) is a Networked Collaborative Virtual Environment system incorporating highly realistic virtual human representations. VLNET allows several users to meet in shared virtual worlds connected through the network. They can communicate and interact with each other, with the environment and with autonomous virtual humans that can inhabit VLNET worlds. The actors possess similar appearances and behaviours to real humans in order to enhance the sense of presence of the users in the environment. The human representation also allows for facial gesture based communication between the users (Thalmann et al, 1997).

VLNET was a joint project developed at both the EPFL-VRlab and the Unige-MIRALab in 1997. It utilised VRML (Virtual Reality Modelling Language) which was programming language developed purely for the creation of virtual environments (see section 4.2).

The advantages of VLNET are:

- The avatar/human representation actually maps pictures of the user onto the avatar to make them as lifelike as possible and enhance virtual interaction.

The disadvantages are:

- VLNET struggles to handle a large number of participants.
- VLNET was originally developed in VRML 1.0 which was replaced by VRML 97 shortly after the projects completion.

4.2 Virtual Reality Modelling Language (VRML)

4.2.1 What is VRML?

The Virtual Reality Modelling Language (VRML) allows users to create three dimensional (3D) virtual worlds which are accessible from the Internet. It provides the basis for the majority of the distributed virtual environments discussed in the previous chapter. It was developed by Pesce & Parisi in 1994. All that is required to access VRML virtual worlds is a VRML browser, this is a plug-in that can be added to both Microsoft Internet Explorer and Netscape web browsers (Nadeau, 1999).

VRML provides an inherently interactive way of presenting information. It provides a method for creating environments that are spatially intuitive and informative enabling methods of communication not possible with traditional HTML (HyperText Markup Language). VRML describes 3D environments using the three Cartesian coordinate references; X, Y, and Z. The coordinates are numbers that place the objects within a 3D grid. X and Y coordinates represent the standard horizontal and vertical references. Z is the third dimension which denotes the objects depth or distance.

A VRML scene has six degrees of freedom as it allows movement along any of the axes, as well as rotation about the axes. In VRML, there is an additional degree of freedom, the hyperlink. Objects in the world can point to other worlds or to HTML documents.

When used in conjunction with VRML it is important to note that the term 3D does not refer to the stereoscopic "depth-enhanced" two dimensional (2D) images as seen in movies which require special glasses. Instead, it means that the 2D visual information on the computer screen is being derived, or rendered, in real time from a logical model of a three-dimensional environment. As a result of this logical model the image can be entirely dynamic. Users can zoom in on points of interest and objects can move, spin, shrink, grow and much more. The user's point of view is unlimited, so they can look up, down, left, right or from any angle or vantage point within the virtual space (Ames et al, 1997).

4.2.2 The VRML Specification

The first VRML specification was released in May 1995 almost a year after it was first developed. This specification lasted a little over a year before being replaced in August 1996 by the second specification. The first specification released had far too many inadequacies shown at a number of conferences by computer programmers. The main criticism of the first specification of VRML was it was released without any provision for interaction. This was subsequently corrected with the release of the second specification. The final specification, VRML97 was released in September 1997. Unlike the first specification update, this time only small changes were made to the language.

4.2.2.1 Overview of the Language Structure

Before comparing the two specifications in more detail, this section provides a brief overview of the format and structure of VRML files. VRML is a language for describing the properties and relations of objects. Conceptually, these objects can be of any form, for example 3D objects, images, sound or text. Within VRML these arbitrary objects are termed "NODES".

Nodes are arranged hierarchically into a construction known as a scene graph which defines an ordered collection of nodes. Within these scene graphs, nodes at a particular position in the graph can affect all the nodes following it. This is used to give certain nodes a set of attributes or properties by defining these attribute nodes prior to the actual object. For example, a cube node could be given texture attributes by defining these nodes in the same scope as, but ahead of the actual cube node. A node has the following properties associated with it:

- **The node type** - This defines what a node actually represents, i.e. a rotation attribute, texture attribute or a 3D object such as a cube or cone.
- **The fields of the node** - A node can contain fields which allow parameters to be defined for a particular type of node. For example, a cube node could have dimensions or a sphere node could have a radius.

- **The node name** - Nodes can be assigned names which prove a very useful feature when running scripts and reusing nodes throughout the code. Nodes being named enable them to be referenced directly from within these scripts.
- **Child nodes** - As previously mentioned nodes are arranged in a hierarchy, thus a node must define itself and all of its children. A node which can have children is termed a group node.

4.2.2.2 Comparing the Specifications

The following series of tables illustrate the main differences between the two specifications within the 5 most relevant areas, selected from the 15 detailed areas within the specification. The data has been collected from the DOCT project white paper (Nadeau, 1997):

FEATURE	VRML 1.0	VRML 2.0/97
Name	VRML = Virtual Reality Modelling Language	VRML = Virtual Reality Modelling Language
Author	Internet community	Internet community
Owner	Internet community	ISO
Release date	1995	1997
Primary rendering system	Interactive	Interactive
Primary application areas	ACAD, scientific, virtual reality	ACAD, entertainment, scientific, simulation, virtual reality
Primary content types	Environments	Environments
Feature summary	VRML 1.0 content may contain multiple shapes, each with geometry, shading, texturing, and transformation specifications. Shapes may be grouped hierarchically, named, and instanced. Light sources may be placed in the environment. Content may	VRML 2.0/97 content may contain multiple shapes, each with geometry, shading, texturing, and transformation specifications. Shapes may be grouped hierarchically, named, and instanced. Light and sound sources may be placed in the environment. Backgrounds and

	include cameras.	fog may be added. Content may include cameras, navigation controls, animations, and interaction controls. Procedural content may be created using Java, and JavaScript. The grammar may be extended via macros.
Comments	VRML 1.0 was briefly in use for creating interactive Web content. With the release of VRML 2.0/97, VRML 1.0 is rarely used any more.	VRML 2.0/97 is in wide use for creating interactive Web content

Table 4.1 – General feature differences between the specifications

FEATURE	VRML 1.0	VRML 2.0/97
Geometry types	Explicit: line, point, polygon Procedural: box, cone, cylinder, sphere, text	Explicit: line, point, polygon Procedural: box, cone, cylinder, sphere, text, elevation grid, extrusion, surface of revolution
Geometry languages	None	Java, JavaScript
Line widths	Always 1 pixel wide	Always 1 pixel wide

Table 4.2 – Geometric feature differences between the specifications

VRML 2.0/97 added the geometry types elevation grid, extrusion, and surface of revolution which is a variation of extrusion. Through embedded Java or JavaScript program scripts, VRML content can procedurally create shapes expressed in one of the built-in geometry types.

FEATURE	VRML 1.0	VRML 2.0/97
Group availability	Yes	Yes
Group hierarchy	Yes	Yes
Group naming	Optional text names	optional text names
Group types	Anchor, group, inline, level-of-detail, separator, switch,	Anchor, billboard, group, inline, level-of-detail, switch,

	transform separator	transform
Instancing types	Per-attribute, per-group, per-shape	Per-attribute, per-group, per-shape

Table 4.3 – Grouping feature differences between the specifications

VRML 2.0/97 restructured the shape grammar to reduce state push and pop, enabling a performance increase on low-end systems. The restructuring added billboard groups and replaced VRML 1.0's separator and transform separator grouping types with the transform grouping type. These are not directly equivalent. VRML 1.0 content using separator groups usually can be translated into VRML 2.0/97 content using transform groups. However, VRML 1.0 content using transform separators is usually not translatable, particularly when those transform separators are used to scope light sources.

FEATURE	VRML 1.0	VRML 2.0/97
Navigation availability	Yes via a common extension	Yes
Navigation constraints	None	Collision, optional gravity
Navigation modes	Examine, fly, walk	Examine, fly, walk

Table 4.4 – Navigation feature differences between the specifications

VRML 1.0 did not support explicit navigation control. A common extension enabled content to specify a global navigation mode. VRML 2.0/97 restructured this mechanism, enabling multiple navigation types to be specified and switched amongst. VRML 2.0/97 also added support for collision detection and gravity.

FEATURE	VRML 1.0	VRML 2.0/97
Interaction availability	Yes	Yes
Input types	Input devices: none User Interfaces: buttons Sensors: none	Input devices: none User Interfaces: buttons, relative locators, valuators Sensors: billboard, collision, level-of-detail, proximity, time & visibility
Response types	Preset (anchors)	Preset, procedural

Interaction languages	None	Animation circuit, Java, JavaScript

Table 4.5 – Interaction feature differences between the specifications

VRML 1.0 included support for click-able anchor shapes. No other form of interaction was supported.

VRML 2.0/97 added a generalized notion of interaction using sensors that could be wired together into an animation circuit without the need to use a programming language. Java and JavaScript program scripts could be written to augment such an animation network and provide more sophisticated interaction control. Several preset interactions, including anchors, billboards, and collision detection are also provided.

Aside from the features listed VRML 2.0/97 also incorporated sound, improved animation, lighting and texturing features. For a full detailed analysis of the difference in features between the two specifications please refer to the DOCT paper (Nadeau, 1997).

4.2.3 Creating VRML

VRML can be written using a text editor such as notepad requiring no financial investment other than the time taken to learn how to write VRML and access to a computer. There are many books available about how to write VRML and there are also some good manuals and on-line tutorials on the World Wide Web. The problem with this method is that hand coding is time consuming. It can be tedious and spotting problems and debugging the resulting code is difficult.

VRML is also a popular world-building tool. These packages allow authors to define worlds graphically and save them as VRML. This process is much faster and easier than hand coding but it is more expensive and often creates VRML files that are complex and large. CAD packages are often used to create 3D models which are then exported as VRML files. Alternatively CAD to VRML converters may also be used. Textures, sound, interactivity and behaviours are then added to the VRML using a text editor. 3D Studio Max is perhaps one of the biggest 3D graphics tools available today

and has a VRML exporter built within the software. However, the files created using 3D Studio Max are generally very large and would require a lot of memory to run.

Once worlds have been created, a syntax checker can be used to check that the VRML code is correct. Software can also be used to optimise the files so that the performance of the VRML is improved. This is often achieved by removing redundant shapes from the code.

Large libraries of VRML objects, textures and sounds are available on the World Wide Web. Some libraries are copyright free, others require the copyright to be credited and others operate on a commercial basis. When selecting objects from a library the units of measurement used are important. Continuity between the files is necessary to keep the files proportionately correct.

4.2.4 An Example of a VRML File

Figure 4.7 is a very simple VRML file which creates a simple cylinder. The following text will breakdown the different lines within the file and provide a brief explanation about what it is doing.

```
#VRML V2.0 utf8
# A Cylinder
Shape {
    appearance Appearance {
        material Material { }
    }
    geometry Cylinder {
        height 2.0
        radius 1.5
    }
}
```

Figure 4.7 – Example of a VRML file

#VRML V2.0 utf8

This line is known as the VRML header and is the first line in every VRML file ever written. **#VRML** tells the computer that the file is a VRML file, **V2.0** states that it conforms to version 2.0 syntax and **utf8** states that the file will use the UTF8 character set which is an international character set standard. It stands for UCS (Universal

Character Set) Transformation Format, 8-bit and encodes over 24000 characters for various languages. Perhaps the better known ASCII is a subset of the utf8.

```
# A Cylinder
```

This line is a comment which is often used to describe the VRML file. Comments start with the number sign '#' and continue until the end of the line. The # tells the compiler to ignore the text which follows it so that a syntax error is avoided.

```
geometry Cylinder {}
```

This is the main node of the file. Nodes describe shapes, lights, sounds, etc. and in this case the node is defining a cylinder. Every node has: a node type (shape, cylinder etc.), a pair of curly braces and zero or more fields. The braces designate where the associated fields start and finish.

```
geometry Cylinder {
        height 2.0
        radius 1.5
    }
```

Fields define a nodes attributes. Both height and radius are fields within the geometry node. Every field has a field name (height), a data type and a default value.

Therefore, within the file the shape node defined that there was going to be a shape used. The geometry node followed which defined what type of shape it was going to be its height, radius and overall size. The appearance and material nodes are also present but have been left blank so the default values will be used.

4.2.5 Browsing VRML

To browse the VRML file all that is required is an internet browser with a relevant plugin. Internet Explorer developed by Microsoft is the most common internet browser used today but there are other possibilities including Netscape, Opera and most recently FireFox. In addition to the internet browser a VRML plugin is required to finally view the VRML files.

There are numerous different VRML plugins and each of them has only slight differences. The plugins all do basically the same job which is to allow the user to

navigate the VRML environments. The way that the plugins differ is through their appearance, the aesthetics of each interface changes. The following list is just some of the available plugins:

- FreeWRL
- GLView
- OpenVRML
- 3Space Assistant
- CASUS Presenter
- Cosmo Player
- Cortona VRML Client
- Blaxxun Contact

The last two plugins on the list are the two most common plugins used today to view VRML. Cortona was developed by Parallel Graphics who also market VRML Pad, the only VRML specific text editor with debugger. Blaxxun Contact is the default plugin for viewing VRML environments running over the Blaxxun Community Platform, the most common Multi-user VRML server.

4.2.6 Multi-User VRML

Creating a VRML virtual environment is the first step when considering using the environment as an interface for a collaboration system. Distributing the environment over the internet is not difficult but when users interact with the environment they will be the only users present. This is because by default VRML worlds are downloaded each time a user chooses to view it. This means that whenever another user wishes to view the environment a separate version of that environment is downloaded to the users' computer.

Multi-user VRML environments work in three ways as discussed in section 4.1.4 on distributed virtual reality. They can either be peer-to-peer, client server or a hybrid server combining the two versions. In general a client server interface is used for multi-user VRML. There already exist multi-user VRML servers that are both commercially

licensed and open source. The following sections discuss the servers available for INTEGRA VICE.

4.2.6.1 VRServer

VRServer follows the client server protocol. It works in two sections, the VRML server executable and the VRML server interface. The executable is written in C++ and takes input from the command line from which the VRML output is generated. The server interface appears to the user as a HTML forms page which is driven by CGI (Common Gateway Interface) scripts. These scripts also execute the vrserver executable.

4.2.6.2 VNet

VNET was developed by White and Sonstein and achieves multi-user VRML using only Java and the EAI. It runs on all of the web browsers with VRML plugins. It has been classed as BOMU (browser-only multi-user) because it doesn't require any special VRML browsers or proprietary technology. It's based on Stephen White's Java classes and his VRML Interchange Protocol (VIP).

4.2.6.3 DeepMatrix

The DeepMatrix System was developed by Geometrek and is another client-server multi-user VRML application. The DeepMatrix system is split into two applications. The first, a server program which runs on the web server and stores the VRML worlds and supporting HTML files. The second is the client application which is realised as a Java applet and uses the EAI (External Authoring Interface) to communicate with any of the VRML browsers discussed in section 4.2.5.

4.2.6.4 Blaxxun Community Platform

Blaxxun Community Platform was one of the first multi user server systems. It was developed by Black Sun in the early nineties and follows the same protocol as DeepMatrix and VRServer. It uses a server to store the VRML environment and then the Java EAI to allow access to the different clients. The Blaxxun community platform is a commercial product but with significant reductions for academia. Because of its commercial use the server software is far more advanced than any of the other servers

discussed. It is also the most common multi-user server on the internet and as discussed earlier, the Blaxxun Contact VRML viewer is also one of the leading browser plugins.

Blaxxun sell the platform as a modular software system, upon which internet-based communications solutions can be produced. Areas of use include E-Learning, Team Workspaces, Interactive TV, Communities, Virtual Worlds, E-Service and Online Customer Clubs.

The Blaxxun Platform offers a comprehensive range of features. Technologies including multi-user and multimedia areas are seamlessly integrated into the platform and enable the development and operation of highly scalable, stable applications. The Blaxxun Platform's features can be accessed through a browser, via HTML, Java or a plug-in. (Blaxxun, 2005)

4.2.6.5 Other Multi-user Servers

There are alternative multi-user VRML servers remaining, however the majority of them have become obsolete as the language has moved on and working examples of them are very difficult to find. There is also little or no documentation available on how to implement these servers. The following list contains examples of these other servers discovered, but unable to test or use:

- Oz Virtual
- Chaco's Pueblo
- Intel's IDMOO
- IDS's V-Realm
- Sony's Community Place Browser

CHAPTER 5

Creating VICE

This chapter describes how the knowledge gained in the previous chapters was used to make informed choices on how to create the VICE system. It examined the considerations made before the final selections were made and the interface written. The chapter starts by fully describing INTEGRA, the system for which the interface is being designed. It then considers how the virtual environment will be modelled and distributed for multiple users. Finally the chapter considers the various options for communication, required for the collaboration that will allow successful concurrent engineering.

5.1 INTEGRA

5.1.1 What is INTEGRA?

As stated in section 1.1, the goal of this PhD was the development of a metaphor based user interface for collaborative work.

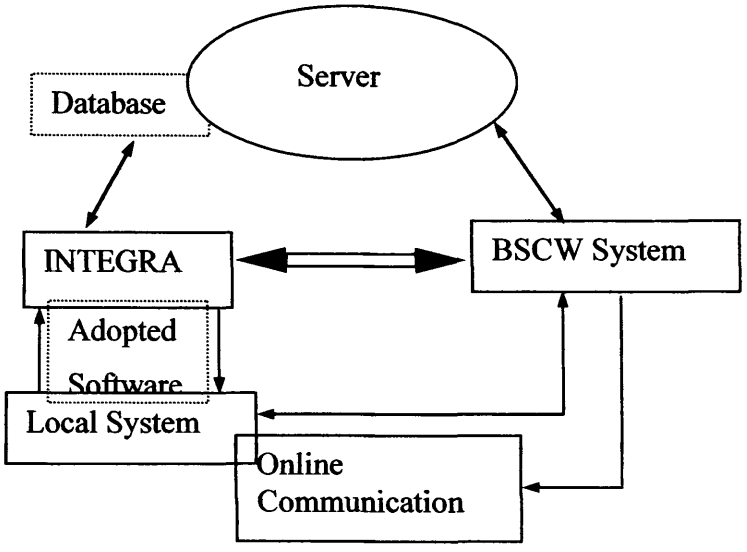

Figure 5.1 - INTEGRA System Architecture

The VICE system (Virtual Integrated Collaborative Environment) had to be created to integrate within the INTEGRA System and become an option for its user interface.

INTEGRA is an internet-based software system that supports the concurrent conceptual design of commercial buildings. Figure 5.1 shows the system architecture of the INTEGRA system. This figure was devised by the INTEGRA research team for the whole system and shows the major components of the system. The original interface designed for INTEGRA was a simple frame based interface which has become standard for most internet based tools or web pages. Figure 5.2 is a screenshot of this interface.

Figure 5.2 – Screenshot of default user interface for INTEGRA

The needs of the construction industry were assessed through a series of interviews with members of all the disciplines involved at the initial conceptual design phase of a construction project. A list of requirements for successful conceptual design was identified using the data gathered in these interviews. The menu bar on the left of the screenshot shows the tools that are available to an INTEGRA user. There have been a number of papers detailing the INTEGRA tools. The following section provides an

overview based on the 'Requirement Capture for Concurrent Conceptual Design'. (Cen et al 2002)

5.1.2 The INTEGRA Applications

5.1.2.1 The Client Brief

The client brief needs to be carefully formulated to provide enough contextual information to ensure a solid foundation for the construction project. The Client Brief tool allows the client to input the brief and all the important information regarding the requirements for the project. These will include the basic requirements of the building being designed such as gross/net areas, gross/net ratio, and car parking space. The client brief is input by the client at the beginning of the project.

5.1.2.2 Uncertainties

The uncertainties tool has been renamed the risk assessment tool because it enables the user to manage the project risk of a building project. It also ensures that the design meets the expectations of the project within the limitations of its capital cost and time. Risks are unavoidable in the construction industry because of uncertainties existing in the financial, economic, political, environmental, design, construction time, site construction, and other factors (Tummala & Bruchett, 1999). These risks can potentially affect the cost, time and quality of a project.

The risk model selected for the INTEGRA system is a tool for risk and uncertainty management. In the model, the risk of a project is regarded as the combination of the risks of all the projects components. It has been found from interview results that the clients, as well as other members of the design team, identify the project risks mainly through experience. Therefore the risk management system can be designed as a knowledge-based system where the client lists activities with uncertainties. The results are then used to predict the effect on the project such as time delay and cost increase. The system then stores the information and uses them as references for similar projects.

5.1.2.3 Cost Model

The system provides an element based cost model which is used to estimate the cost of the building design. The cost model is based on numerous recent observations that 80%

of the project cost is contained within 20% of the most expensive items. These items usually cost roughly the same within the class of similar projects. The reference prices for element costs of office buildings can be found from previous project reports which quantity surveyors or cost managers publish quite regularly on journals, such as *Building* (Building, 1994). The cost of a project can then be predicted from the values of these elements.

5.1.2.4 Drawing Tool

The drawing tool allows users to sketch design ideas using a pen-based tablet system called WACOM INTUOS. Once a design option is finally accepted by the design team, the designer then transfers this solution into a more formal drawing either by sketching with the use of WACOM INTUOS or by drawing with the aid of Architectural Desktop software. To produce a 3D image of the design option, the designer can sketch or draw different perspectives of the design and then integrate these into a 3D view of the design using Photo Vista Reality Studio software.

5.1.2.5 Constraints

The constraints tool is used for constraints input and checking. The constraints for a building design generally come from members of the design team during the design process and are stored in the system. The client brief also states some of the constraints for building design. These defined constraints can then be used for constraint checking. The constraint checking of design options is designed to be both graphics based and text based. Some of the parameters such as gross area, net area, and car parking can be calculated directly from the drawing of a design option. Constraint checking of other parameters such as project cost, floor to ceiling heights, and number of stories, will use a text-based approach.

5.1.2.6 Design Rationale

The design rationale is an area of the INTEGRA system where information is stored that can be used by the design team to track design decision and to detect conflicts. Generally, design rationale includes the reasons behind a design, the justification for it, the alternatives considered, the tradeoffs evaluated, and the arguments which led to that decision. Design Rationale can be recorded in free-text basis or in structured ways. It

is easy for users to enter info using the free-text based method but the information can not be detected and used by the computer. The alternative is to use the structured methods which allow the input information to be reused but may cause some difficulties for designers to describe their opinions. The design rationale in the INTEGRA system will initially use a semi-formal method where the data are input as free text base and indexed by keywords.

5.1.2.7 Database

The database of the INTEGRA system consists of public and private sections. The public section stores the design information that the owner is willing to share with all members of the design team. The information stored in the private section however, can only be accessed by its owner. The public database is located within the public workspaces of the INTEGRA server, and the private section kept on the user's PC or the private workspaces of INTEGRA server. The system provides a file manager for both public and private database sections that can show the history and relationship of the files. For instance, the file manager records that several files store different versions of the same design option. It also records information about the name of the creator and editors of a file as well as the production time.

5.1.2.8 Communication Tool

The communication tools are necessary for concurrent engineering to occur. At project meetings, members of the design team present their constraints that need to be considered by the Architect and Structural Engineer at a later stage. To achieve a good design option, the Architect and Structural Engineer must be able to communicate their design ideas to other members and make necessary corrections on the basis of feedback. The communication tool in the INTEGRA system allows a member of the design team to communicate with a single member, as well as with several members at the same time. The latter can be regarded as a project meeting.

In the INTEGRA system, files of design options and information are transferred between members of the design team generally in one of two ways. One method is by putting the files in the public database and sending an email to relative members, which states the names and locations of available files. This method is suitable for passing the

complete version of design options or information that a designer would like to share with all members. Another way of information exchange is by attaching the files to an email message. This is suitable for a designer to send change notifications to specific members. The system also provides an environment for members of the design team to discuss rough ideas or design options via the Internet. During online conversations, attendees sketch their ideas on an electronic whiteboard and make comments on the sketches from others. With this method, members of the design team can exchange their ideas more efficiently and quickly.

5.2 Modelling a Virtual Environment

Section 5.1 looked at INTEGRA and exactly what the conceptual design tool involves and should give a better understanding and what is required from INTEGRA VICE. Communication is of obvious importance for concurrent conceptual design to occur. This is because, as stated in chapter 2, constant collaboration is imperative for successful concurrent engineering. INTEGRA VICE is an interface that bases the INTEGRA system within a communication system. Using VICE as an interface will allow users to be in constant communication whilst accessing the remaining tools. This means that instead of having to select the communication tool as they would with the default frame based system. The users will always be capable of some form of communication from the default interface default screen. The benefit of this is a far more streamlined design process and the interface should allow a more intuitive use of the software.

VRML (discussed in the previous chapter) is the best and most advanced way of creating a virtual environment. There are also alternatives to hard coding the virtual environment. Many software packages exist which will allow the user to create a three dimensional drawing and then export it to a VRML file for use with a VRML enabled browser. Using such software applications allows users to literally draw the office using a CAD (Computer Aided Design) package rather than trying to create it using geometry and code which is the case when hard coding VRML files.

In theory this would save so much time as very little actual programming would be required so the two main packages were initially considered to carry out this 3D design. These two packages stood out above all others when considering 3D design and they were AutoCad and 3D Studio Max. Both packages provided a VRML exporter which would allow the created 3D environments to be traversed using a VRML enabled browser. However both packages suffered from the same problems, discussed in the following sections.

5.2.1 3D Studio Max

3D Studio Max is a modelling/animation package developed by discreet. It uses an open architecture to encourage program additions and features. These features allow 3D Studio Max to continuously evolve to suit the needs of the individuals using it. It is also this open architecture that lead to the VRML exporter that enables the drawings to be converted into virtual environments.

3D Studio Max was developed from 3D Studio for DOS for the win32 platform. It is now in its 7 incarnation on this platform and each time a new release is added it often involves incorporating many of the tools and add-ons that have been created by other users due to the open architecture. The key features of the newest release are the extensive animation tools, and UV mapping tools which allow the lighting of the object to be fully controlled by the user. There is still the traditional modelling tool for object creation and they have now added MaxScript a scripting language to help add functionality to any tools created. There is also now an almost inexhaustible array of materials available for rendering, contained both in the software and through online libraries.

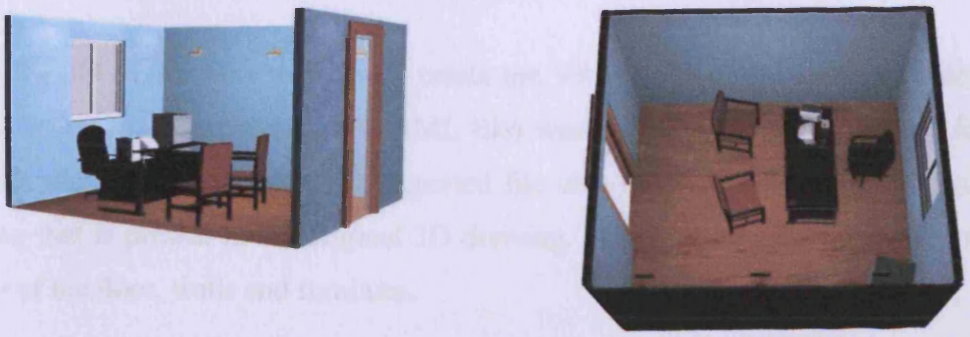

Figure 5.3 – Views of first VICE Office Created Using 3D Studio Max

Figure 5.3 shows a couple of views of the first three dimensional personal office developed for VICE using 3D Studio Max. The personal office is where most of the interface will be housed and also where the majority of the collaboration and concurrent engineering will occur. The office is relatively simple with only the basic furniture but it contains a great deal of detail including different lighting, and rendered floors and furniture. 3D Studio Max handles really excels at the rendering of objects. To render objects the user simply has to select the material from the materials list and click on the plane to be rendered. Therefore achieving such a design is relatively simple when using 3D Studio Max because there are many tools that allow the render of different planes. This office was then exported using the VRML exporter and figure 5.4 shows a screenshot of this exported VRML office.

Figure 5.4 VRML Personal Office Exported from 3D Studio Max

Using the 3D Studio Max exporter to create the VRML file was reasonably quick and simple however the exported wrl (VRML file) was extremely large, especially for use across a distributed network. The exported file also did not include the rendering or lighting that is present in the original 3D drawing. This accounts for the difference in colour of the floor, walls and furniture.

When using the exported environment and trying to traverse the office the computers performance was drastically reduced and the movement appeared jerky and imprecise.

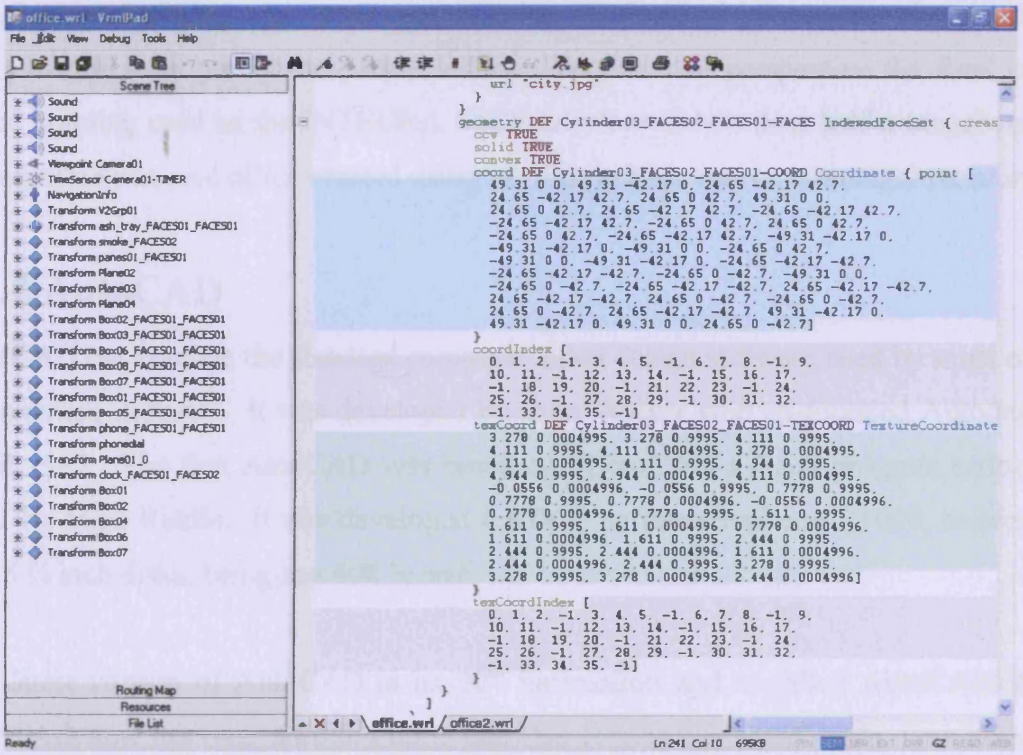

Figure 5.5 – Screenshot of the Exported Code

Figure 5.5 shows a screenshot of the code from the exported VRML file viewed using VRMLPad. Studying this file gives a better understanding of the increased size of the exported files. The texCoord field within the file designates the various coordinates of the points on a plane. The texCoord field above shows some 60+ coordinates with multiple decimal places to create a relatively simple shape. The shape created may appear in the environment as purely a cylinder but the exporter has not used the cylinder function from within VRML. Instead the exporter automatically computes the exact position of every part of the object and creates the corresponding coordinates using the texCoord function to recreate the object.

The exported file size is so extreme because the software does the same with every object present in the environment. There is also no in-lining taking place which means that none of the created objects are re-used. For example the cylinder in the above example is for one of the desk legs and for the other three legs there is essentially the same sized texCoord field beneath. If you compare that to the cylinder created in

section 4.2.4 a simple cylinder which takes just two lines to create, the reason for the exaggerated size of the exported files becomes more obvious.

A full office building was never created using 3D Studio Max because the size issue was obvious from purely the private office. To put it in perspective the final entire office building used as the INTEGRA VICE interface is less than half a megabyte yet the exported personal office created using 3D Studio Max was over a megabyte alone.

5.2.2 AutoCAD

AutoCAD has become the standard computer aided design software used by most of the construction industry. It was developed by John Walker who co-founded Autodesk in April 1982. The first AutoCAD was based on MicroCAD a CAD program written in 1981 by Mike Riddle. It was developed for DOS software and cost $1000, coming on two 5 ¼ inch disks, being just 40k in size.

The latest version of AutoCAD is its 20th incarnation and is called AutoCAD 2006. AutoCAD now has over 6million users and this version has many tools, now allowing values to be entered and options selected using purely the cursor rather than the command line of old. AutoCAD now comes on a CD-ROM and is over 600 megabytes in size which is fifteen thousand times greater in size than the original release from 1982.

Figure 5.6 shows a three dimensional office building created using AutoCAD. There was no attempt made at creating the more detailed personal office drawn using 3D Studio Max because a problem with the exported file was predicted at the onset. Therefore a relatively simple office building shell was used. It is also more difficult to quickly create a drawing as detailed with AutoCAD as it is used more for technical drawing where as 3D Studio Max is more commonly used for graphic design.

As expected, when converting the AutoCAD file the same exporting problem occurs and although the geometry is less advanced, the exported file is still far greater in size than necessary and will lead to poor usability of the system. Another, perhaps bigger problem with the exported AutoCAD files is their reduced functionality. Creating

animations to allow simple things like the doors to open has to be hard coded into the created VRML file after it has been exported. This proves very complicated when the coordinates have been defined by AutoCAD and there is no commenting in the file to designate which node is doing what.

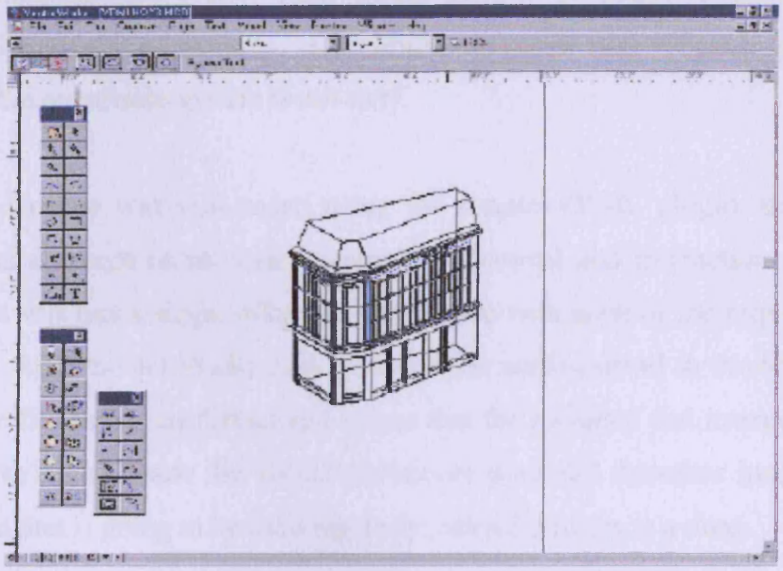

Figure 5.6 – AutoCAD Drawing that will convert to VRML

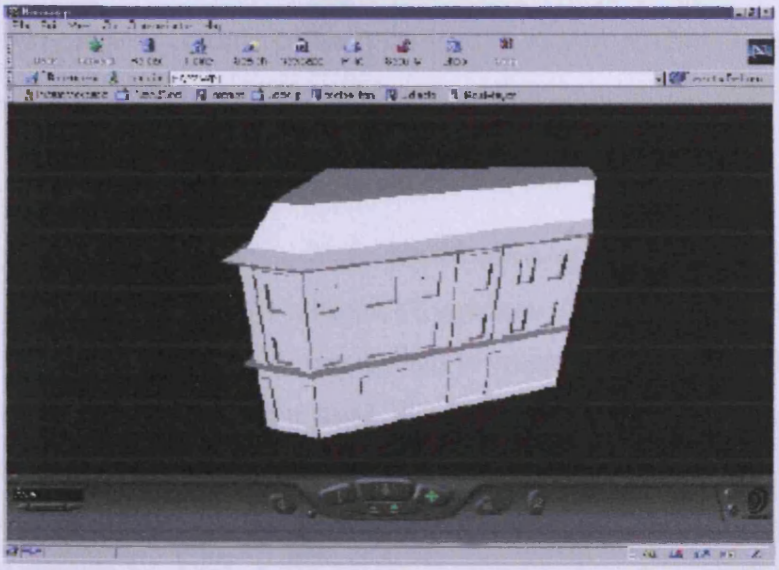

Figure 5.7 – Same AutoCAD Drawing converted to VRML

Figure 5.7 shows the exported office building viewed through Netscape using the cosmo- player VRML plugin discussed in section 4.2.5.

The exported AutoCAD file looks relatively simple and is two tone in appearance. There is also the issue of the internal space. Defining internal offices is not too difficult but providing furniture that appears as furniture should in the virtual environment is difficult to achieve. The difficulty stems from the time consuming nature of creating these furnishings in AutoCAD. They would need to be added to the exported file through hard coding at a later date. This would not prove very efficient as scaling and transposing the coordinate system is not easy.

The exported office was also tested using the simple VRML plugin, this time with Netscape and although initial tests showed the traversal and interaction speeds to be adequate this was just a single office building space with none of the required detail of an interface. Like the 3D Studio Max exported file adding detail to the office building increased the file size even further and meant that the traversal and interaction became slow and jerky. This made the virtual experience poor and therefore inadequate for a user interface that is going to be used regularly, often for hours at a time.

5.2.3 VRML

Modelling the virtual environment using simple text editor is far slower than either of the previous two methods considered but it is the only way to produce a small file sized environment. Keeping the size of the wrl down is of paramount importance because of the networks distributed nature.

The only way to create advanced aesthetic virtual environments when hard coding the VRML is to use a trial and error method of creation. Geometry has to be used to calculate the position of the coordinates required to create the object and then using the debugger built into VRML Pad the created object can be viewed. This is an extremely slow method when compared to using the exporter from 3D Studio Max or AutoCAD but the created file is far more usable, more than 10 times smaller in file size. If the object created needs to be changed, the VRML is edited and the file is debugged again.

5.3 Multi User Servers

Once the VRML virtual environment has been created a server has to be used to allow the VRML the functionality of multiple users, a prerequisite of a successful user interface for concurrent engineering. Multiple user VRML is far more advanced than a single user environment. Trying to create the projects own multi-user server was considered however this was a task which would require too much time and far greater programming knowledge to be developed. As discussed in section 4.2.5 there are available servers that provide multi-user VRML environments. The servers considered initially were VNet, DeepMatrix and Blaxxun Community Platform.

5.3.1 VRServer for VICE

VRServer was available from Tenet at the start of the project and during the initial testing phase of available multi-user VRML servers. However, Tenet published the following on the VRServer website shortly after testing had begun: "*This program is no longer supported by Tenet and no future development is planned. Use it at your own risk!*" Shortly after this announcement the Tenet website was shut down so using the software became an impossibility as there was no download or documentation available.

5.3.2 VNet for VICE

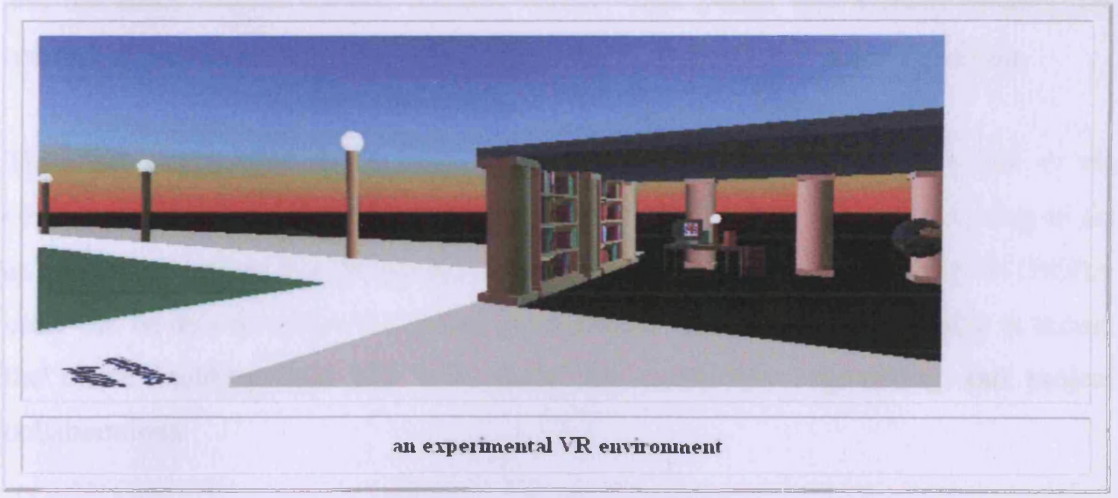

Figure 5.8 – VNet Vritual World

There are various virtual worlds on the web using the VNet server to handle the multi user capabilities. During testing of these worlds there was never more than one user present in any of the worlds.

This meant that establishing how good the multi user software was proved difficult. Figure 5.8 shows a screenshot of one VNet virtual world visited on the vrmLab at Streamer (vrmLab, 2005). Because of the inability to physically test how well the server operated with multiple users logged in VNet was not developed for the VICE system any further.

5.3.3 DeepMatrix for VICE

DeepMatrix developed by Geometrek is still widely used by multi-user VRML worlds. There are numerous worlds available on the web and many of them are often visited by multiple users. Geometrek have also set up a hub to act as a link to all of these available worlds.

DeepMatrix is an open source VRML multi-user server. Therefore using this server for VICE would be very cost effective. However there are a number of issues that made DeepMatrix an impractical choice for the VICE multi-user server. Firstly the documentation available to aid the development of a DeepMatrix Server is inadequate and the email support service appears faulty. This means that actually creating the server and transferring VICE's VRML files will be difficult and time consuming.

The other problem is that to use the Geometrek DeepMatrix system a link to the environment has to be posted on the Geometrek hub. If the environment is going to act as the user interface for INTEGRA then it will need to be secure. Only INTEGRA users can be able to access the virtual environment. If random users are able to access the office building then this will hinder the concurrent engineering and project collaborations.

5.3.4 Blaxxun Community Platform for VICE

Blaxxun Community Platform is a commercial multi-user VRML server but the cost to academia is greatly discounted. After reviewing the alternative servers Blaxxun became the obvious choice. The server software is discussed in detail in chapter 6 where the final interface is described. The main advantage of using the Blaxxun server is that it is totally adjustable but at the same time works as simply as plug and play. There is no unnecessary hard coding required which will save the research project valuable time.

5.4 Communication Software

The main aim of the project was to develop an alternative metaphor based interface for INTEGRA and it also specified the need for full user collaboration. This means that the interface needs to include the necessary software to allow as many different ways to collaborate as available.

The default INTEGRA system uses Microsoft NetMeeting as its communication tool and can be selected from the frame based interface. There are many alternative communication tools available on for purchase or development. The following sections discuss some of the options considered for the INTEGRA VICE system.

5.4.1 Microsoft NetMeeting

As it was already being used by the default interface and is an established communication tool, Microsoft NetMeeting had to be considered. The software is a free add-on to Microsoft Windows, so all Windows users will have access to it. However users from any other operating system will not have it, therefore it is perhaps not practical to be the main communication tool of choice.

5.4.2 Video Chat ActiveX 1.0

Video Chat ActiveX 1.0 was developed by Viscom Software and is a relatively simple application available for free download. It allows you to send video through a web cam and audio through a microphone to other internet users. The application can be

embedded into any software that is written in a programming language which supports ActiveX such as Visual Basic, C++, Delphi and many more.

The problem with using this software in VICE is that it only promotes one on one interacting so another alternative application would be required for the multi-user video conferencing that is needed for VICE. Therefore it is impractical to use.

5.4.3 ICU Conference 1.48

ICU Conference 1.48 was developed by AdriaComm and is another free tool that promotes communication over the web. Unlike Video Chat ActiveX 1.0 this tool does promote multiple user video conferencing. The software can handle many simultaneous users so it would be a viable solution for the boardroom conferencing requirement. It also provides application sharing such as interactive whiteboards.

The problem with using ICU Conferencing 1.48 as the communication tool for VICE, is that it is a separate application. It can not be embedded into a web site. This means that the application would have to be started from VRML somehow and the users would have to learn how to use it. This reduces the overall effectiveness of VICE because the system will require more knowledge/skill and less intuition to use it. For this reason it is not a practical solution for VICE.

5.4.4 Video Conference 1.0

Advanced Software Logic created Video Conference 1 and released it in 2002. It is another free application available for download. The software is used purely for sending and receiving video over the web. Like Video Chat ActiveX 1.0 Video Conference 1.0 can be embedded into web pages. This is the most practical solution for VICE as HTML can be linked to VRML which will avoid the issue of starting applications from VRML.

The main problem with incorporating Video Conference 1.0 into the VICE system is that it does not provide the functionality required for other communication. There is no whiteboard application or shared desktops etc. Incorporating a single application that

handles all of these communications will help reduce the required knowledge of the users when using the system.

5.4.5 Macromedia Flash Communication Server

Flash has been around for a long time however the communication server is a relatively new addition to the family. Flash communication server utilises Flash player, a free plugin from Macromedia that allows the broadcast of flash files within internet browsers. The Flash plugin is one of the most common in the world. As of June 2004 Flash Player 6 was present on more than 94% of internet accessible workstations.

The Communication Server runs alongside the website server and handles all necessary communication. Using Flash, tools can be written which interact with hardware such as the video camera and microphone to create conferencing applications.

Flash can be embedded in any website so like Video Conferencing 1 and Video Chat ActiveX 1 the communication tools can be created specifically for VICE. This gives the flexibility to fulfil all of the needs and requirements of the system. Having the designated server also seems to speed the communication up drastically. Using just standard broadband, there is very little delay and the audio comes through fluidly without distortion.

5.5 Conclusion

A number of decisions were made during the early stages of creating VICE. Each choice was made for specific reasons as stated above. The virtual worlds will be written in hard code using the VRML language and VRML Pad. The finished VRML environment will use the Blaxxun server to distribute it and Flash will be used to write the communication applications necessary for collaboration.

CHAPTER 6

INTEGRA VICE

This chapter describes the INTEGRA VICE system. It contains many screenshots showing how the interface looks and explains all of the tools that can be interacted with and used. The chapter explains what and why choices were made. The chapter also details snippets of code however including the full system code in the thesis, even in an appendix would be impractical as it may reach over a thousand pages.

6.1 INTEGRA VICE – The Programming

6.1.1 System Architecture

Figure 6.1 shows a simplified diagram of the system architecture. This diagram is more specific to the user interface and does not include all of the applications from the INTEGRA system (Taylor et al, 2003). VICE follows the client-server protocol as discussed in section 4.1.4.2. However the interface uses two servers which run simultaneously.

One server uses the Microsoft Internet Information Services encoding whilst the other is a Macromedia Flash Communication Server as shown in the centre of the diagram. The reason two servers are required is that the Flash server is solely for handling internet communications that have been written using the Macromedia Flash language. BSCW is a third server which is a document handler used by INTEGRA for managing project documents. The Video conferencing tool, whiteboard and messenger are coming off the flash communication server because they are written and handled with Flash. Other tools such as Outlook and Microsoft Office are linked to the client computer because these tools would be on that machine and not distributed.

The advantage of using the macromedia language is that the communication applications can be run just using a standard flash player. A designated server is important due to the high bandwidth required by such communications. These flash communication applications are discussed in more detail in section 6.3. Everything else is handled by the Internet Information Server including the Multi User VRML and the BSCW software.

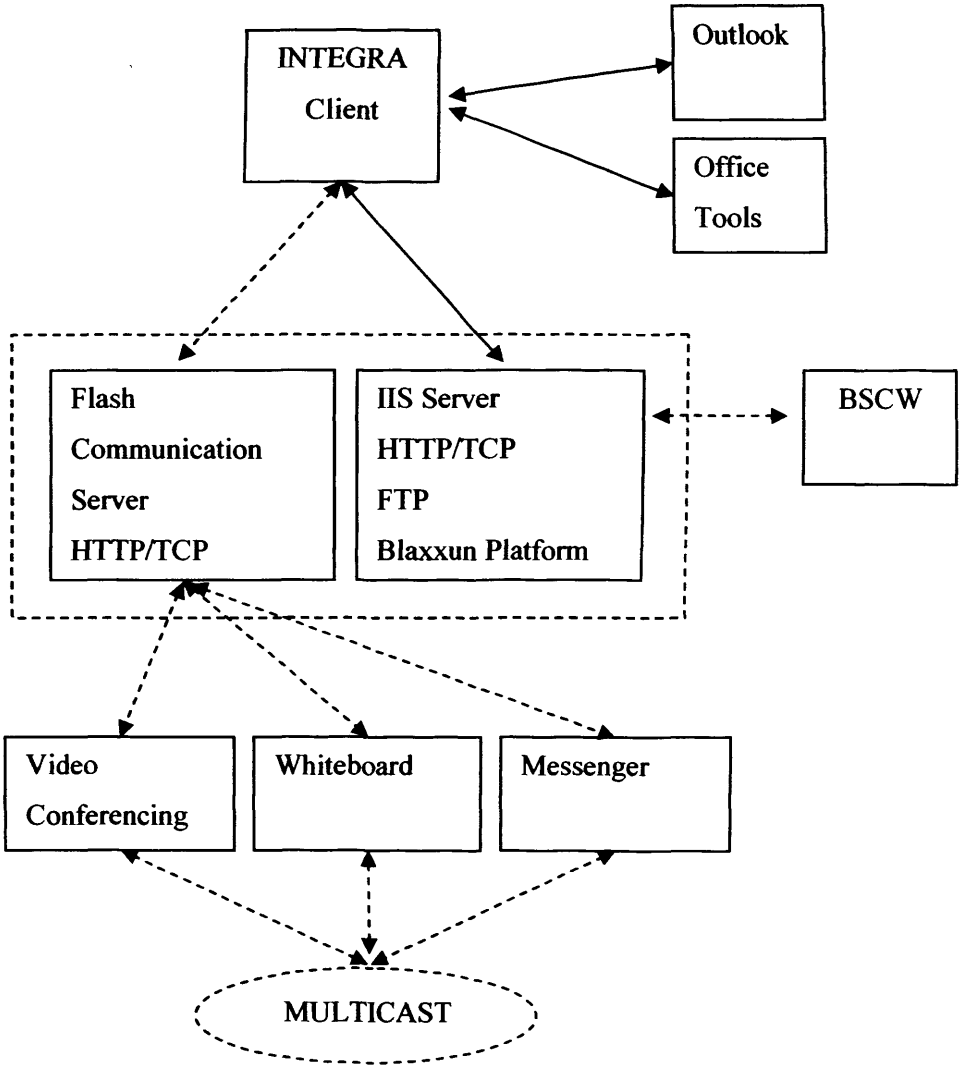

Figure 6.1 – System Architecture

Figure 6.2 shows a screen shot as seen from the eyes of a user sat at his desk within the VICE interface. There are two further users present within his office. The two users have not specified avatars and therefore appear as the browser's generic avatar. This scene is described in more detail further on in this chapter.

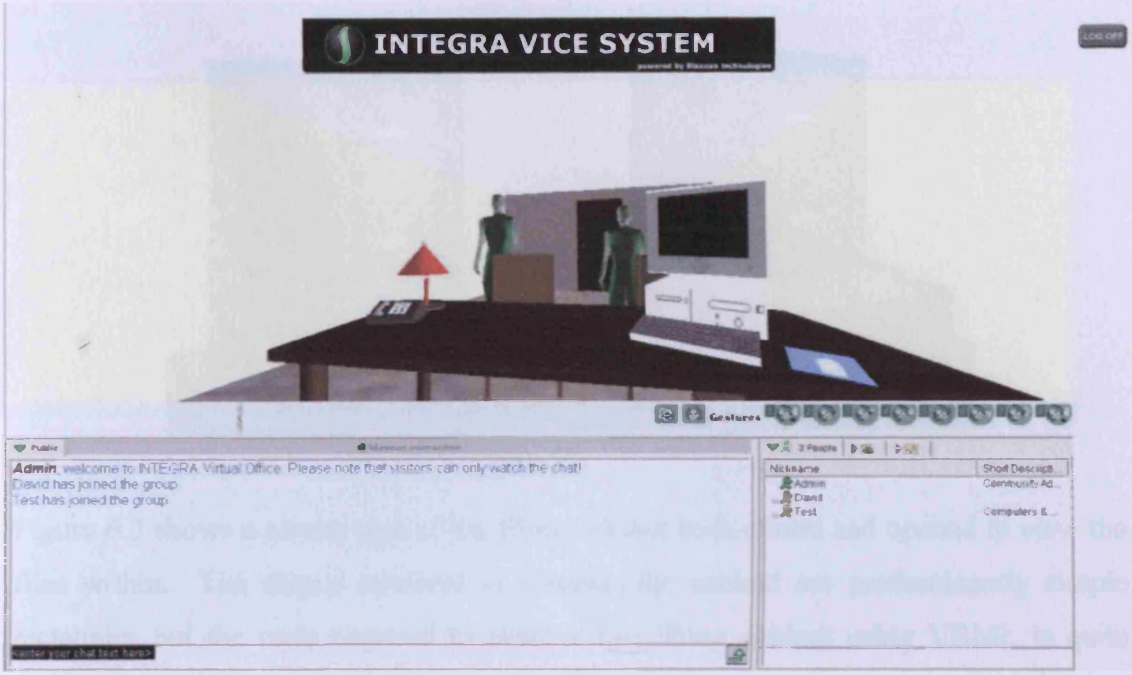

Figure 6.2 – Screen Shot from INTEGRA VICE

6.1.2 VRML

As discussed within the previous chapter it is not possible to use a graphics package to create the virtual world from a graphics point of view and then export it to a VRML world. Therefore the entire virtual office building has been created using a VRML text editor and geometry.

This meant that creating the virtual office building would take far more time and effort as each point had to be calculated and defined to produce the shapes required to make any of the objects. A good example of this is the geometry required to create the filing cabinet alone. Each office contains a filing cabinet which is the virtual metaphor for a real filing cabinet thus containing access to all of the projects files. Interacting with the filing cabinet connects the user to the document handler and enables them to view, edit, add and delete any files that they have access rights to.

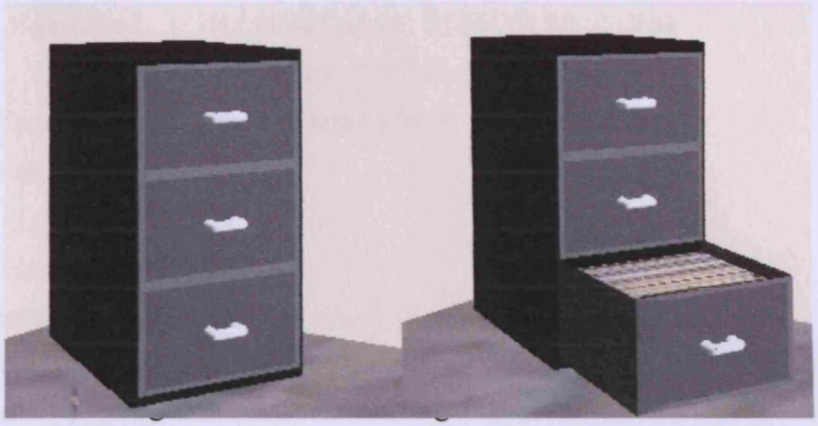

Figure 6.3 shows a screen shot of the filing cabinet both closed and opened to view the files within. The shapes involved in creating the cabinet are predominantly simple rectangles but the code required to achieve this filing cabinet using VRML is quite considerable. An example of this code is shown below:

```
#VRML V2.0 utf8
Group {
children [
Shape { appearance Appearance {
material Material { diffuseColor 0.5 0.5 0.5 }}
geometry IndexedLineSet { coord Coordinate {
point [-1.5 1.0 2.5, 1.5 1.0 2.5, 1.5 1.0 -2.5, -1.5 1.0 -2.5,-
1.5 -1.0 2.5, 1.5 -1.0 2.5, 1.5 -1.0 -2.5, -1.5 -1.0 -2.5]}
coordIndex [ 0 1 2 3 0 -1,  7 6 5 4 -1, 0 4 5 1  -1, 1 5 6 2 -1,
2 6 7 3 -1, 3 7 4 0 ]}}
Shape { appearance Appearance {
material Material { diffuseColor 0.25 0.25 0.25}}
geometry IndexedFaceSet { coord Coordinate {
point [-1.5 1.0 2.5, 1.5 1.0 2.5, 1.5 1.0 -2.5, -1.5 1.0 -2.5,
-1.5 -1.0 2.5, 1.5 -1.0 2.5, 1.5 -1.0 -2.5, -1.5 -1.0 -2.5
-1.5 0.75 2.5, 1.5 0.75 2.5, 1.5 0.75 -2.5, -1.5 0.75 -2.5,]}
coordIndex [7 6 5 4 -1, 0 4 5 1  -1, 1 5 6 2 -1, 2 6 7 3 -1,
3 7 4 0 ] solid FALSE }}]}
#Files Picture
```

```
Anchor { children Shape {appearance Appearance {
material Material { diffuseColor 0.25 0.25 0.25}
texture    ImageTexture {url "Officefiles.jpg"}
textureTransform TextureTransform {
translation      0.0 1.0
rotation 1.571  }}
geometry IndexedFaceSet {
coord Coordinate {point [ -1.5 0.75 2.5, 1.5 0.75 2.5, 1.5 0.75
-2.5,-1.5 0.75 -2.5,]}
coordIndex [0 1 2 3 0 -1]
solid FALSE}}
#Links to file which will open program
url   "startapp.html"
parameter "target=leftFrame1"}
Shape { appearance Appearance {
material Material { diffuseColor 0.35 0.35 0.35 }}
geometry IndexedFaceSet { coord Coordinate {
point [ -1.5 1.0 2.5, 1.5 1.0 2.5, -1.5 -1.0 2.5, 1.5 -1.0 2.5,
-1.4 0.9 2.5, 1.4 0.9 2.5, -1.4 -0.9 2.5, 1.4 -0.9 2.5,]}
coordIndex [0 4 5 1 0 -1, 0 4 6 2 0 -1, 2 6 7 3 2 -1, 3 7 5 1 3]
solid FALSE}}
```

Trying to explain all of the different nodes and various VRML programming techniques that are present in the above code would be pointless but just viewing the code should adequately emphasise the complexity of hard coding a virtual environment. The code above only creates a single drawer of the filing cabinet and only the drawer itself without the handle seen on the front. Showing the full code would take far too many pages and is unnecessary.

As well as the complexity of the code itself, the structure has been altered to make it more compact for the purposes of this chapter. The code as typed within a VRML file would look very different and this is demonstrated in figure 6.4. Figure 6.4 shows a screen shot of a VRML file within the VRML text editor VRML Pad. The Scene Tree is like the document map tool in Microsoft Word and allows quick traversal of the file.

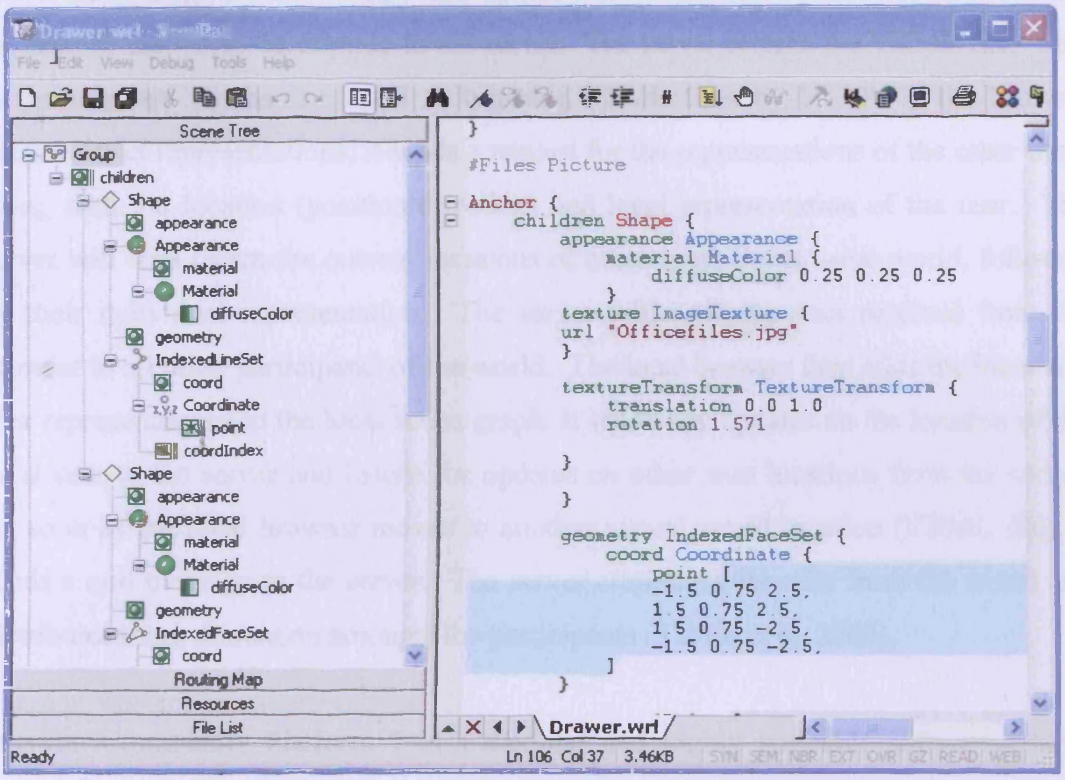

Figure 6.4 – Image of VRML file within VRML Pad

The main difference between these two examples of code is the layout. It is good practise to use a tabbing and spacing system, because it enables the programmer to see where the bracketing system used in VRML starts and ends. This system is present in figure 6.4 but not in the typed code. The reason for using this system is to help when files need debugging after syntax errors have occurred. This file also contains the multiple user extensions required to allow multiple users to interact with each other within the virtual environment.

6.1.3 Multi-User VRML

To work as a successful user interface the virtual office environment has to be multi-user, i.e. capable of handling multiple users interacting within the same space at the same time. This requires a designated server that can handle communication and interaction from all of the users.

VICE uses the Blaxxun Community Platform to handle these communications (See 5.3.4). To interact with the server, the VRML file needs certain protocol extensions to

distribute user representations and also update their locations. The browser first sends a request for the world description to the server. The server returns the VRML file. This is the standard mechanism used to transmit VRML files by HTTP. If the browser supports user representations, it sends a request for the representations of the other users along with the location (position/direction) and local representation of the user. The server will then return the current locations of other users in the same world, followed by their individual representation. The server will send the data received from the browser to all other participants of the world. The local browser then adds the incoming user representations to the local scene graph. It sends any updates on the location of the local user to the server and listens for updates on other user locations from the server. As soon as the local browser moves to another virtual world location (VRML file), it sends a quit message to the server. The server eliminates the user from the world and distributes this information amongst the participants (Taylor et al, 2005).

Blaxxun Community Platform 7 is a modular and highly scaleable software-system comprising a multitude of communication and interaction components. This software fully supports the multi-user VRML operation, administration and provides usage tracking of the virtual world. It is an open system that supports all the relevant standards to enable 3D multi-user interaction and it works together with the blaxxun client (web browser plug-in). (Blaxxun Interactive [1], 1998)

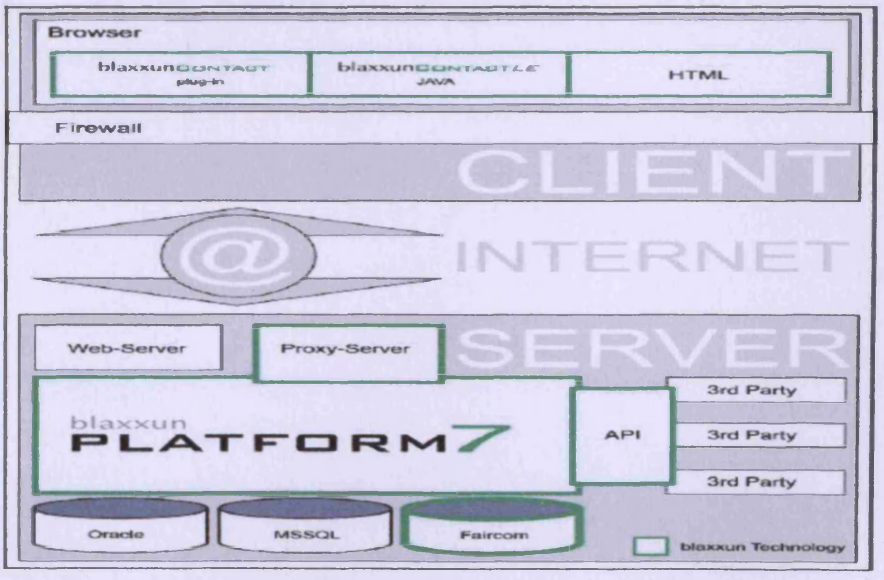

Figure 6.5 – System Architecture of Blaxxun Community Platform

In addition to this, blaxxun client provides a way of routing VRML events (Blaxxun Interactive [2], 1998) to all visitors of a multi-user place. In this way, all users in the virtual world can view animations and interactions that are triggered by one user. This is a necessity for true multi-user interactivity. Without this functionality when one user opens a door the rest of the users will not see anything happen. The user will just appear to walk through a closed door. Figure 6.5 shows the architecture of the community platform software provided by Blaxxun. Figure 6.6 shows a screenshot of the server console adapted for INTEGRA VICE.

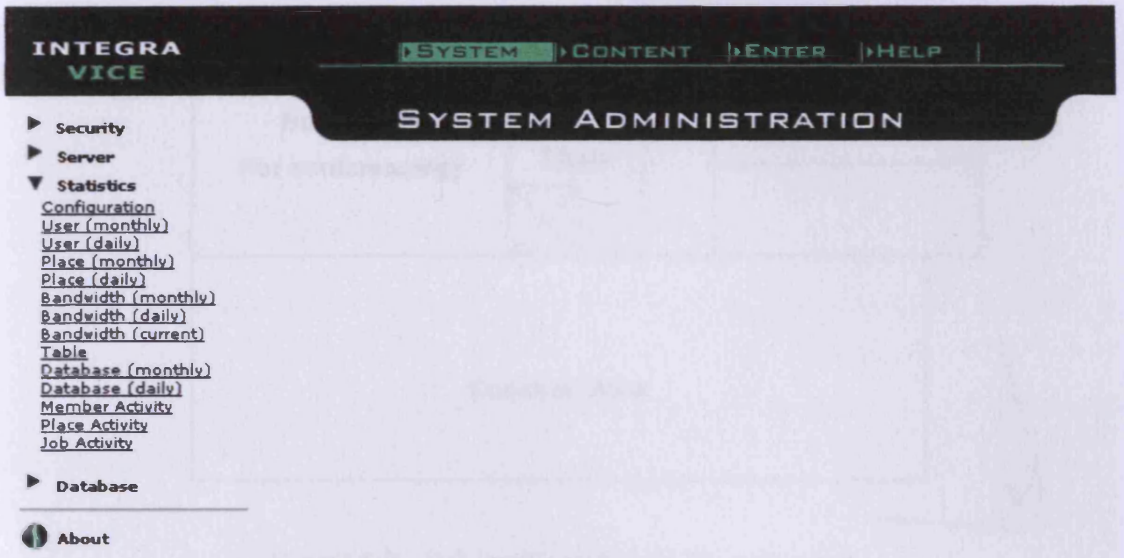

Figure 6.6 – The INTEGRA VICE Multi-User VRML Server Console

6.2 INTEGRA VICE – The Virtual Office Building

6.2.1 Office Building

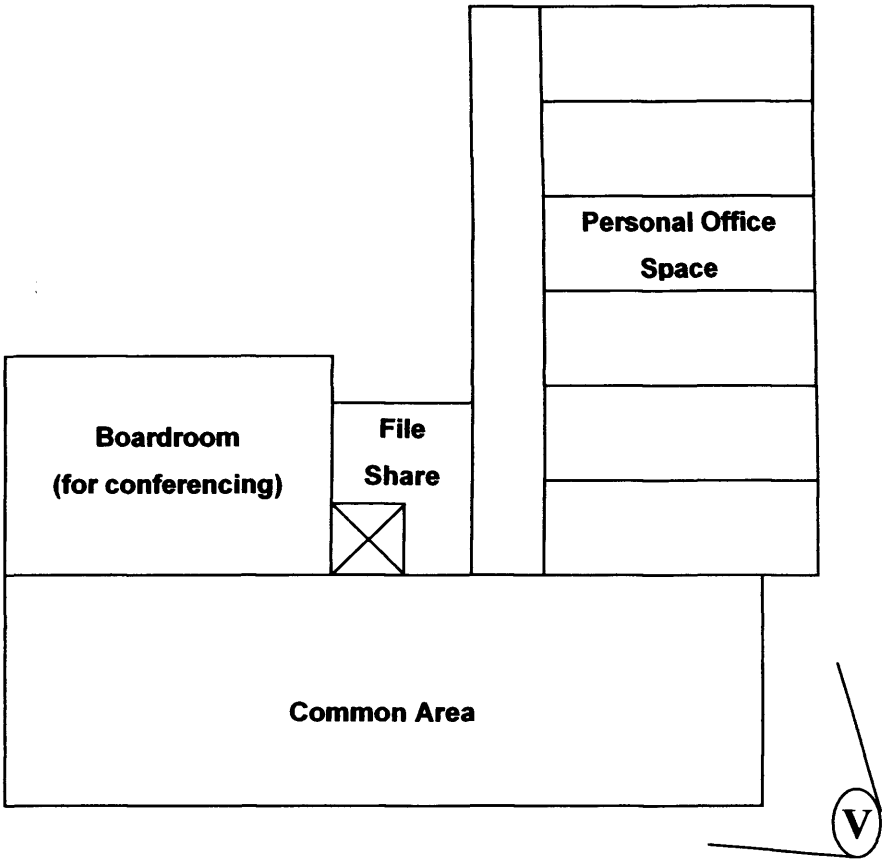

Figure 6.7 - Schematic of Single Floor Layout

The virtual office building has several floors. Each discipline is based on a different floor with access to each floor being via a lift (elevator). A new office building is set up for each design project with the relevant number of floors that are required. Figure 6.7 shows a schematic layout of a single floor within a virtual office building (Taylor et al, 2004). Keeping the layout simple is in keeping with the need for a fast accessible system in which the object is not to achieve total reality, but to provide sufficient information for a comprehensible user interface. Each project team member receives a private office on being registered as a system user for a given scheme. From this office most of his/her work and communication is carried out. The other areas available for exploration are the communal areas, file share rooms and conference rooms. The (V) shows the position of the avatar who's view is depicted in figure 6.8. The view is from outside the three dimensional office building and shows how the building looks from an elevated view.

Figure 6.8 – Virtual Office Building

6.2.2 Private Office Space

The private office is where all users start upon log-in. This is also where most of the work is carried out and most of the interactions occur. From his/her office, the user is able to access all of the system's applications; video conferencing, file share, email, whiteboard etc.

Figure 6.9 – Private Office Space

Figure 6.9 shows a screenshot of the private office space. The office contains standard office furniture and many of the objects allow interaction with the user. These objects are also metaphors that allow the users to interact with them in the same way they

would within their own office at work. This is important as it keeps the system simple allowing users to use the system instinctively.

Figure 6.3 shows the filing cabinet which is situated within the personal office of all users. This three dimensional object is the metaphorical representation of the electronic file storage used by the project management system. Opening the filing cabinet and clicking on the files launches the designated file handling system which, in the case of the INTEGRA system is BSCW.

Figure 6.10 shows the view of the user from his/her desk. It is from the same viewpoint as the screen shot shown in figure 6.2 but without the multiple users present. While "seated" at the desk the user can access most of the available tools.

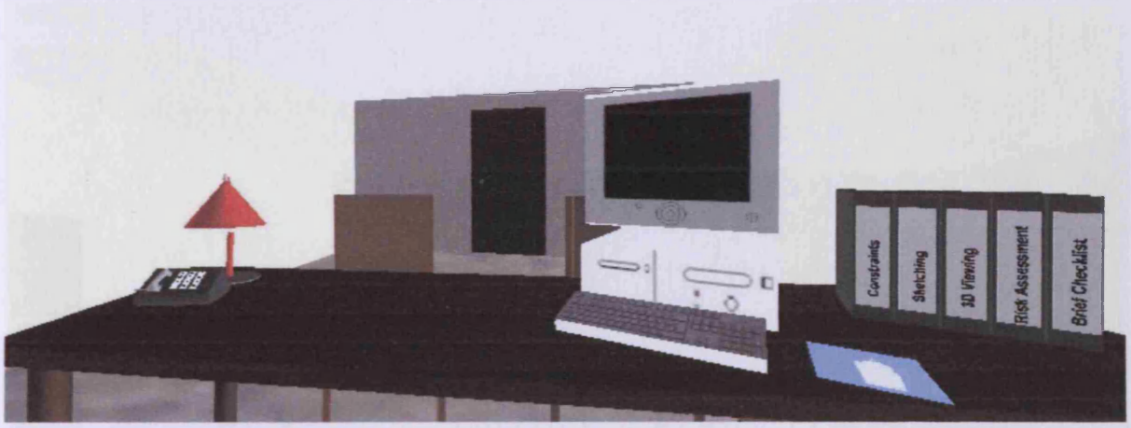

Figure 6.10 – User's View of Desk and Tools from Chair

The computer gives access to email and agenda systems such as Outlook. These are activated by merely clicking on the computer. In a similar way the phone allows the sending of an instant message to any user, perhaps requesting a face to face meeting. One on one video conferencing software is accessed via clicking on the guest chairs.

The two chairs opposite the desk are where other users sit when visiting the office. The user can see the avatars until the video conference replaces this view. The office door opens and closes and there is a whiteboard on the wall to the user's right which can be used to start the interactive whiteboard application. Clicking on the whiteboard allows

all users present to use the application. The filing cabinet is out of view to the users left.

6.2.3 Communal Area

The communal area is an area within each floor that acts like a common room. When the user leaves his/her office they will enter the communal area and will be able to interact with any other user situated in the room. This will allow for general discussion and informal meetings between co-workers. This area will be helpful to the user if he/she has a small problem that they need help with but do not want to bother someone who is busy. It is also from the common room that access to the boardroom and other floors within the building is gained.

Figure 6.11 – Communal Area

These further floors holding the remaining project members (engineers, architects etc.) are all accessed via a lift (elevator) situated in the middle of the back wall of the common room (see section 6.2.6). Figure 6.11 shows the ground floor common room (client floor). The avatar stood in front of the viewpoint is of another user of the system. The users can use default chat to communicate or they can initiate a video conference.

6.2.4 File Share Room

There is a general file storage area that can be found on each floor within the building. Any user with security clearance can access files placed in this room, however, the files kept in the private file stores will be exclusively accessible to the room owner. The general file store room also contains a three dimensional filing cabinet albeit a slightly larger version. When interacting with the general file store filing cabinet the user will be directed to the BSCW document handling system currently being used by the INTEGRA system. Figure 6.12 shows a screenshot of one of these general file storage rooms from the communal area door.

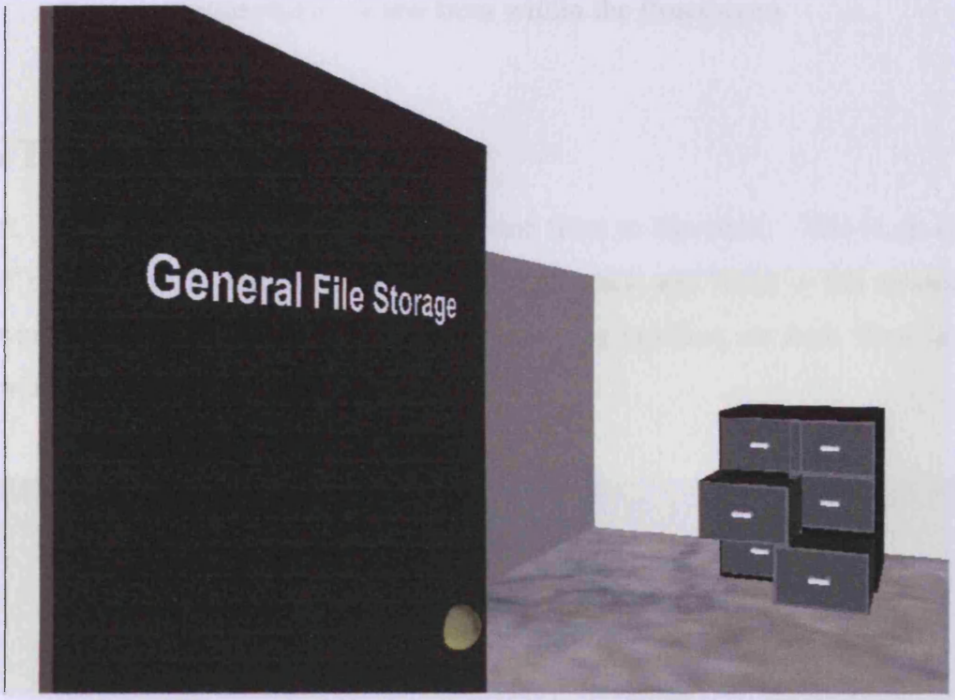

Figure 6.12 – General File Storage Area

6.2.5 Conference Room

The conference rooms contain large tables around which many users can sit. The rooms have whiteboards and the multi-user video conferencing takes place in them. Users simply need to access the room before the conference is due to start in order to be included in the video conference. The conferencing software (i.e. the Flash application) is started by clicking on the large boardroom table in the centre of the room. Figure 6.13 shows the conference room with another avatar present.

Figure 6.13 – View from within the Boardroom

6.2.6 The Lift

The lift (elevator) is used to transport from one floor to the other. This is an optional method of transport as the user may simply right click and beam to his chosen floor from where ever he is in the building. The starting position on each floor is in the appropriate communal area.

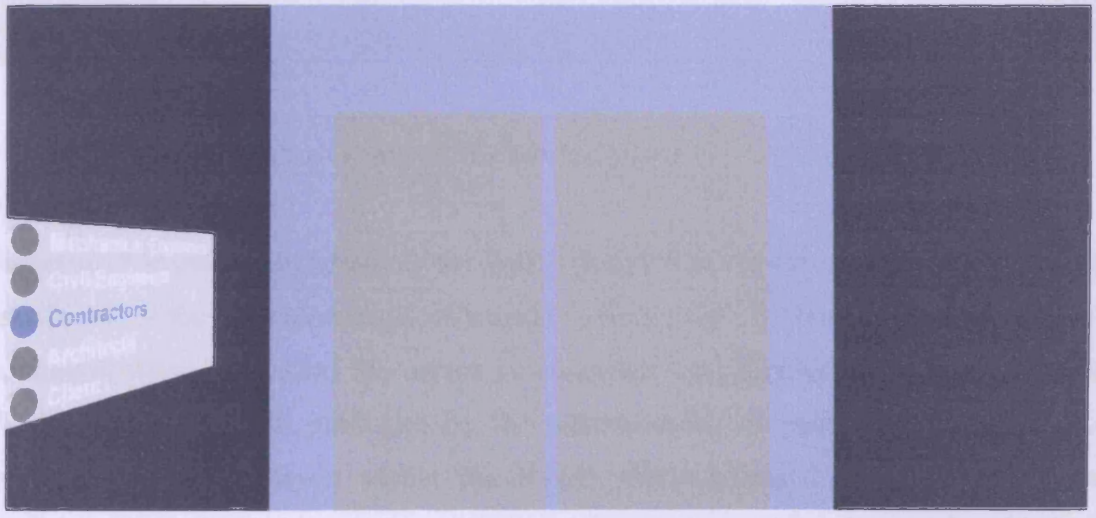

Figure 6.14 – View from within the lift

Figure 6.14 shows a view from within the lift. The user is on the contractor's floor but the lift doors are closed. Clicking on the doors initialises the animation that will cause them to open.

6.3 INTEGRA VICE – Communication Tools

6.3.1 Flash Communication

After considering all of the possible communication tools that were discussed in section 5.4 Macromedia Flash in combination with the Flash Communication Server was chosen.

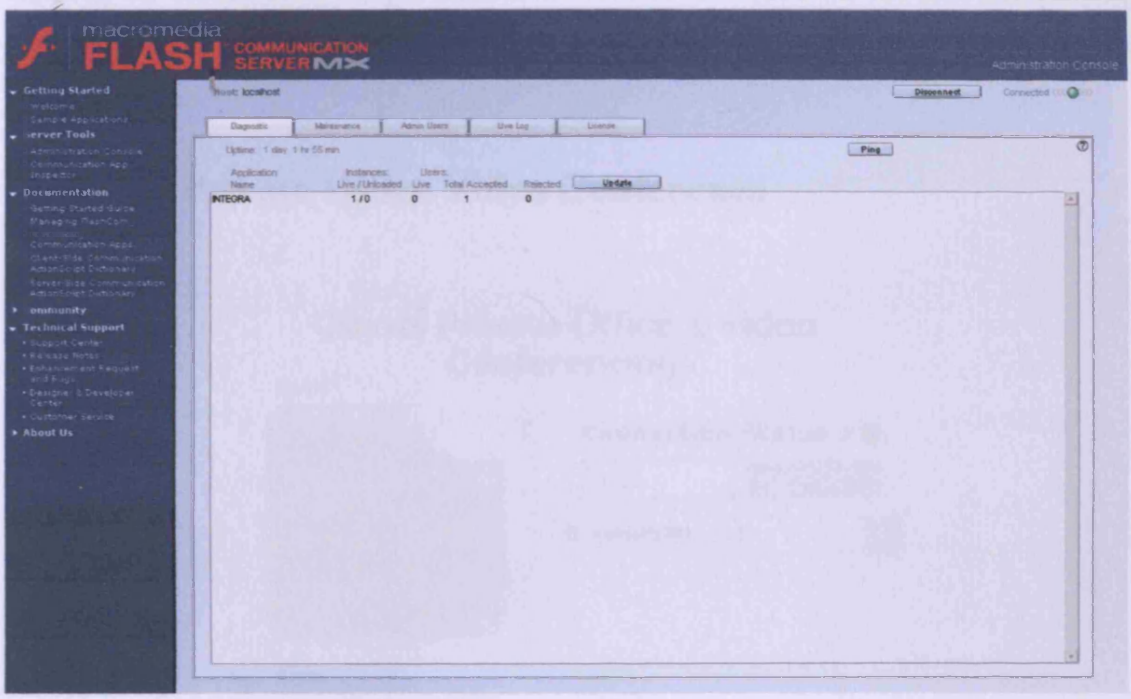

Figure 6.15 – Screenshot of the Flash Communication Server Console

Figure 6.15 shows a screenshot of the flash communication server console. The server handles all of the communication calls made to the server. Each time a user instigates a communication application the server is contacted and handles that communication. Flash is an application published by the Macromedia company. Flash works as a 'window' that is displayed within the HTML environment. The flash author has complete control of the window and animations are usually displayed. Flash is animation software used to develop interactive graphics for Web sites as well as desktop presentations and games. Flash can now, with the users consent, capture video and audio from hardware devices attached to the workstation. This allows real-time communication to take place, enabling the development of video and audio

conferencing software. The Flash Communication Server runs on any workstation connected to the web and requires a standard web server to run alongside it. INTEGRA VICE utilises this method of operation as can be seen from the system architecture shown in figure 6.1.

Writing the communication tools may be more challenging than purchasing ready made software but it gives the author complete control over appearance and usability. Keeping the functionality simple reduces the risk of confusion for the user and ensures the intuitiveness of the VICE system.

6.3.2 The Private Office Video Conference

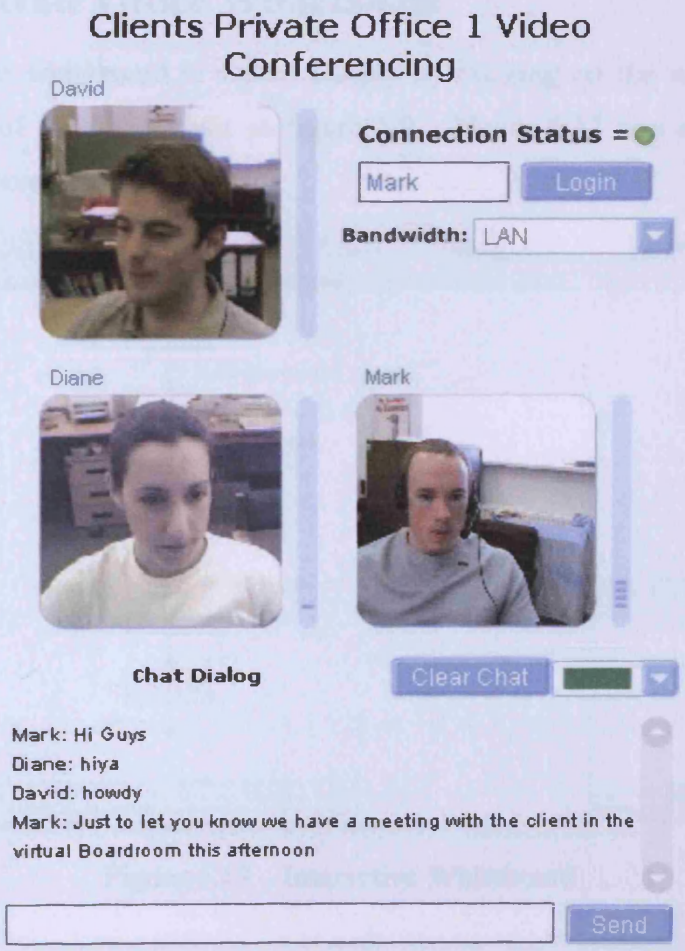

Figure 6.16 – Private Office Video Conference

As discussed in section 6.2.2 when clicking on the guest user chairs a video conferencing application is started. Each private office has its own completely separate version of the video conference application. Figure 6.16 shows a screenshot taken of the video conferencing software being used during the user evaluation. This is a video conference taking place in the office 1 of the client's floor as shown by the application title.

The application has been kept relatively simple with the only options available to the user to type in the text box, alter their bandwidth or clear the text window. Using the text chat is unnecessary as audio chat is available but most users during the evaluation still actively chose to text chat as well.

6.3.3 The Private Office Whiteboard

The private office whiteboard is started simply by clicking on the whiteboard on the wall, to the left of the users chair in figure 6.9. Figure 6.17 is a screenshot of the interactive whiteboard.

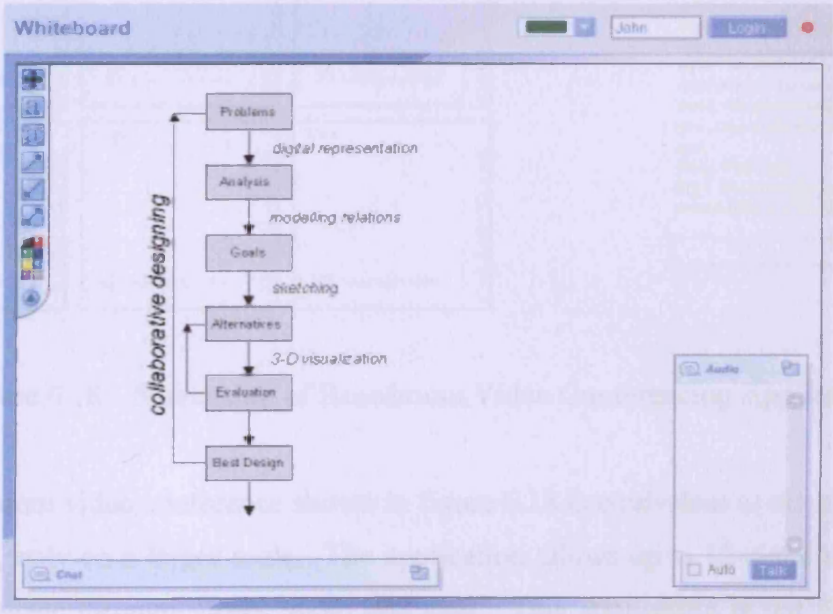

Figure 6.17 – Interactive Whiteboard

Like the personal video conferencing tool each private office space has its own personal interactive whiteboard. The main purpose of the whiteboard is to allow users to create and alter sketches whilst discussing them. Such a capability is essential to speed up

design processes and promote concurrent engineering. The whiteboard provides text and audio chat but no video conference. Using video conferencing on top of the whiteboard made the application too detailed and 'busy'. The whiteboard functions are standard sketching tools that would be available with software such as Paint.

6.3.4 The Boardroom Video Conference

Figure 6.18 – Screenshot of Boardroom Video Conferencing Application

The boardroom video conference shown in figure 6.18 is equivalent to the private office conference, only on a larger scale. The application allows up to 12 simultaneous users to communicate through video, audio and chat. This application is not limited to 12 however keeping it to 12 ensured fast and fluid interactions at the available bandwidths. Aside from the extra video windows the remaining tools and buttons are the same as the private video conference tool.

6.3.5 Microsoft NetMeeting & Outlook

The default INTEGRA tool uses NetMeeting to handle its communication. Therefore the VICE system also incorporated the software in case of problems with the flash communication server and for users who are more comfortable continuing to use it. NetMeeting is accessible by clicking on the virtual telephone on each user's desk.

Microsoft Outlook has been set up as the default agenda/email client for the system. Clicking on the computer will start this application. This function can be changed relatively easily by renaming the process called so that it reflects the alternative application such as lotus notes.

Calling applications from VRML is not an easy task. In fact VRML is not able to handle the JavaScript that is required to start the process which will run an application. This caused a lot of problems during the implementation of the VICE system before a clever loop was devised that allowed this inadequacy of VRML to be side stepped. The VRML virtual environment was placed within a very small HTML frameset.

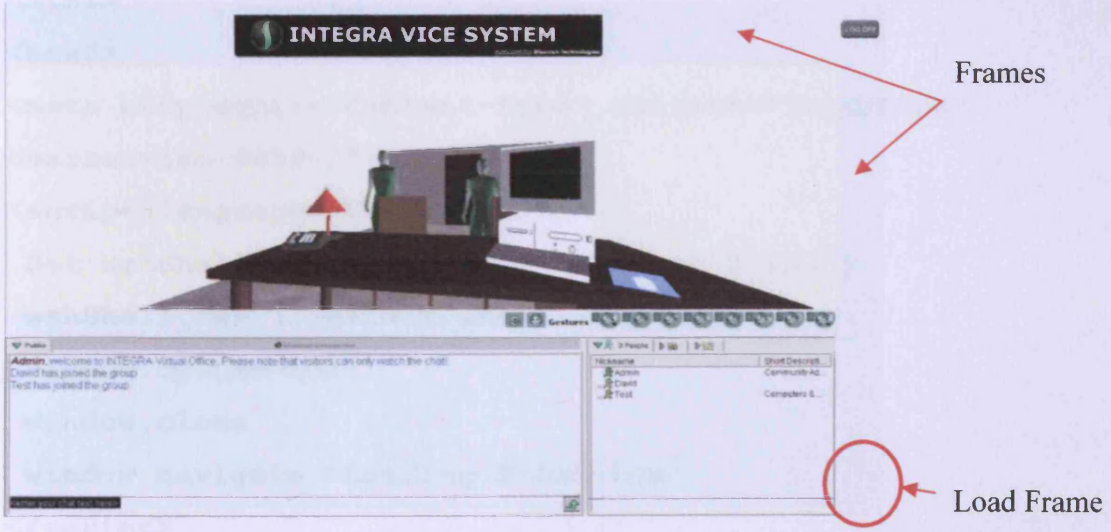

Figure 6.19 – Screenshot Demonstrating Frameset

Figure 6.19 shows a screenshot of the interface with the HTML frames labelled. The Load Frame can not be seen by the user as it is a small box coloured the same as the other frames which make up the border and provide the system title and log-off button. It is this load frame that allows process calls. When writing in VRML you can use

VRMLScript which is very limited and also link to HTML files using a simple url function. You can not utilise VBScript, a far powerful script language required to initiate processes and start applications. The following code shows just how simple the side step was:

```
#Links to file which will open program
url  "startOE.html"

parameter "target=loadframe"
```

The url function was used to call a html file called "startOE.html". Then within this HTML file some VBScript was written to initiate the process and start the application. The frameset was used to stop the open VRML environment from being replaced by the opened HTML. The startOE.html file was called to the "loadframe", the 'invisible' frame specifically created for starting applications.

The startOE.html file contains the following code:

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
<script language="VBscript">
 Set wshShell = CreateObject("Wscript.Shell")
 wshShell.run ("outlook.exe" )
 window.opener=null
 window.close
 window.navigate "Loading Frame.htm"
</script>
</head>
<body bgcolor="#F7FFFA">
</body>
</html>
```

Using this sidestep drastically increases the power of VRML because it enables the use of VBScript and can link that VBScript to virtual objects allowing them to act as buttons.

CHAPTER 7

User Evaluation

This chapter explains the user evaluation carried out to ensure maximum usability of the system and gain constructive feedback on various different types of people's usage of the system. The chapter first explains why user evaluations are needed in interface design and then describes the two evaluations carried out. The chapter also analyses the results from the evaluations. This was another of the objectives which needed to be fulfilled in order for the overall project aim to be achieved.

7.1 Why Run User Evaluations?

User evaluation is essential in the production of any product that is going to be used by other people. Without completing these evaluations the overall usability of the system may be substandard. There are various ways of completing a user evaluation ranging from specific information about very few users to less specific information about a lot of users. For example, video taping a single user using the system would allow a detailed analysis of exactly how he/she interacted with the system. Where as better testing of a product would involve multiple simultaneous users filling out questionnaires after finishing the evaluation.

The user evaluation created to analyse VICE was chosen because of the following constraints and requirements:

- Time – there was limited time to complete the evaluation.
- Cost – the costs of completing the evaluation arose due to the need for hardware and a location to use.
- Location – the location of the evaluation would affect the number and type of evaluator present.

- Access – there was access to just a select number of individuals from only certain demographics.
- Analysis – specific analysis was required to enable improvements to be made.

7.2 Initial User Evaluation

The aim of the initial user evaluation was to test the navigation and interaction within the virtual world as well as test the collaboration tools written in flash. The initial evaluation was carried out by research students and undergraduates during May 2004. Ten individuals took part in the evaluation at three separate times, working in threes or fours. The evaluation was carried out in a designated office where computers had been set up correctly with the necessary hardware and software. Initial testing also provided the opportunity to identify and correct any minor glitches/faults with the interface.

During this initial testing period each user was given 5 minutes to freely traverse the virtual office building before being handed a set of instructions which would lead them through the building and ensure that all the areas that required testing were visited.

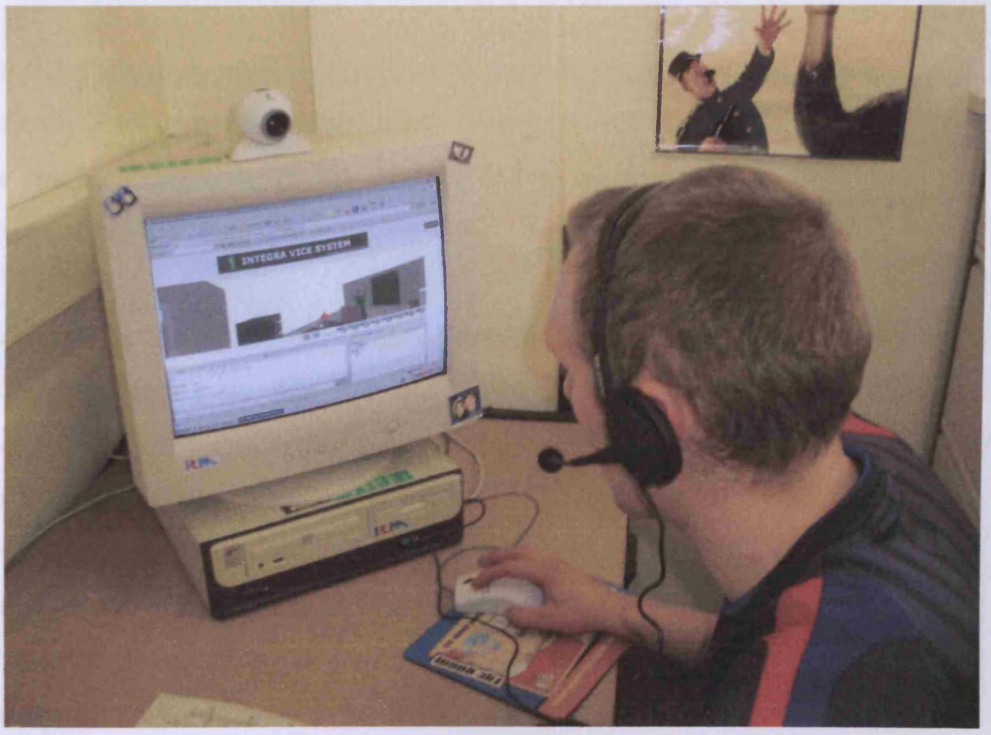

Figure 7.1 – User Evaluating the INTEGRA VICE System

After completing the testing each user completed a simple questionnaire which asked them to evaluate certain areas of the system on a scale of 1 – 7, 1 being very poor and 7 being excellent. Figure 7.1 shows a photograph of one of the students testing the VICE system. Another benefit of completing the initial user evaluation was that the evaluation identified inadequacies of the evaluation method itself. It allowed revisions to be made to improve the questionnaire and the manner in which the evaluation was carried out. The main problem highlighted by the initial evaluation was the need for further comment boxes throughout the questionnaire to allow more freedom to the evaluator for his/her thoughts.

7.3 Final User Evaluation

The final user testing was carried out with 20 users from two main demographics. Ten of the users were engineers with extensive experience in engineering design whilst a further ten were students from the university. The students did not have a full understanding of certain aspects such as concurrent engineering but were from a generation of computer users who were fully adept at using a mouse to traverse a virtual space.

The initial evaluation made it clear that to provide results that could be looked at more analytically a set of tasks was required. Using predefined tasks ensured that each user had a similar experience of the system and more importantly, it ensured that they experienced the entire system.

The testing was carried out in groups of two or more but usually involved three or four users simultaneously so that the multi-user aspects of the system could be fully appreciated. It was also tested on machines no slower that Pentium III and with a minimum standard broadband connection of 512kb/s.

The final questionnaire was split into five sections:

- Virtual Office building – looks at the aesthetics and ease of use of the virtual metaphor
- The Tasks – ease of completing the eight tasks

- Private Office Space – looks at the interactions within the private office space
- Flash Communication Applications – evaluates the flash collaboration tools
- Overall Effectiveness of the User Interface – in particular, how VICE improves project management during the early phases.

Each section also had a comments box which allowed each evaluator to make any comments he/she desires. To analyse the results of the questionnaire, each section will be considered in turn and the average ratings considered. The charts represent the average score for each question from the two demographics with 4 as average and 7 excellent. Using a seven point scale promotes a more honest response. It has been noted that during evaluations it is very rare that a user will award the top or bottom mark on a sliding scale. Consequently, the more options available to the user, the better the opportunity to analyse the resulting data.

An example of the questionnaire is in the appendix C accompanied by a example of the completed questionnaire from the user evaluation. Figure 7.2 has been taken from the questionnaire to better demonstrate how the scaling system has been used.

The Virtual Office Building	Very Poor	Poor	Less than adequate	Adequate	Good	Very Good	Excellent	Not Applicable
• Ability to move around the virtual environment.	1	2	3	4	5	6	7	n/a
• Interaction with other users.	1	2	3	4	5	6	7	n/a
o Gestures	1	2	3	4	5	6	7	
o Chat	1	2	3	4	5	6	7	n/a n/a
• Adequacy of presentation.	1	2	3	4	5	6	7	n/a
Any other comments:								

Figure 7.2 - Example of the Questionnaire

7.4 User Evaluation Results

7.4.1 The Virtual Office Building

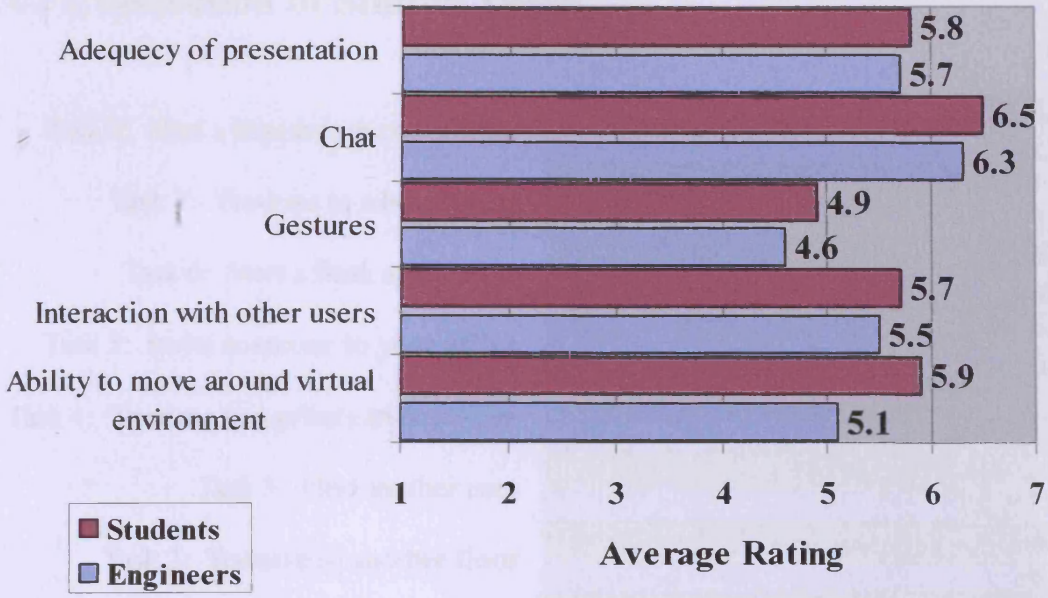

Figure 7.3 – Results from Section 1 of the Questionnaire

Figure 7.3 clearly shows that the evaluators found the chat system particularly effective in both cases. This is not particularly surprising as most computer literate people would have experienced and used a text chat system such as messenger before. Even without prior experience, any user could adapt their knowledge of word processors to make use of the text chat system. Throughout all of the questions in this section of the questionnaire, the students have a higher average response, especially in the case of the user's ability to move around the virtual environment. This is also evident in the comments made by the engineers, an older demographic. The engineers found walking around the virtual environment very difficult at first but most adapted reasonably quickly and no user rated this as less than adequate or below. The reason for this difference is probably due to the younger users experience with computer games which often use a similar view point and method of traversal.

The lowest mark in this section of the questionnaire was gained by the avatar gestures. These gestures are a feature of the Blaxxun software rather than the VICE system and

are not obvious to the user. Both demographics were happy with the presentation of the virtual environment with some commenting positively on the systems realism.

7.4.2 Completion of Simple Tasks

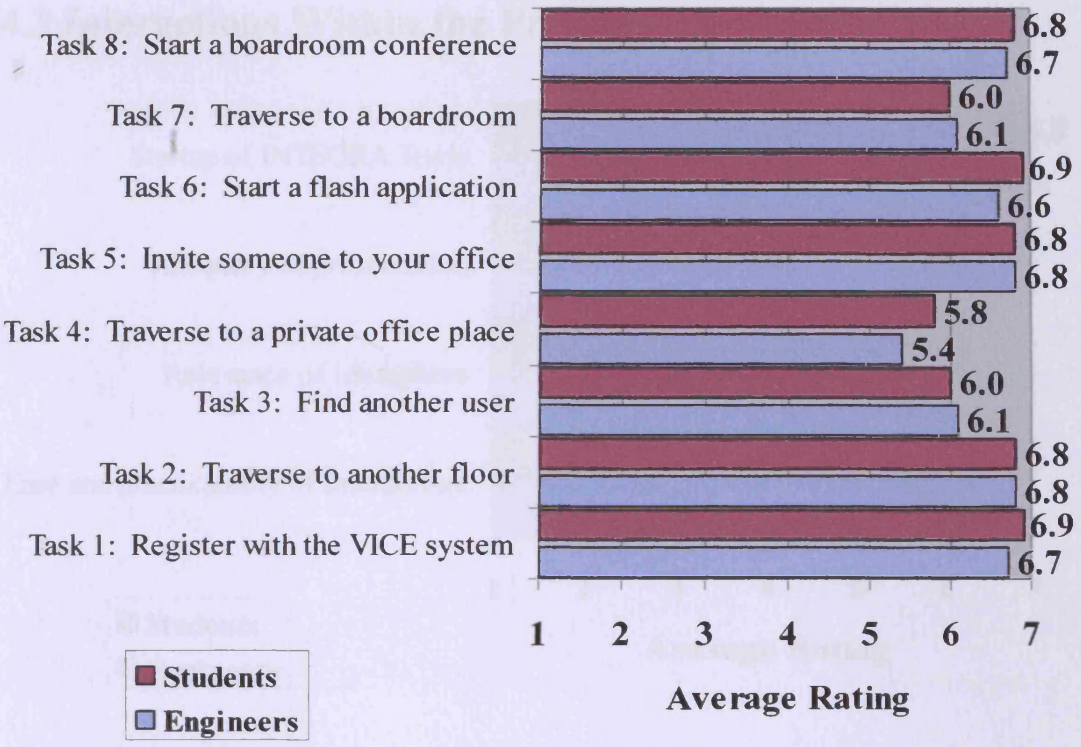

Figure 7.4 – Results from the Task Section of the Questionnaire

It is clear from figure 7.4 that the eight tasks were completed relatively easily by all of the users. This was as expected and the only average less than 6 (very good) was task 4, traverse to a private offices space. From the comments made, in particular by the engineers, this was due to the doors closing too quickly, before they had time to walk through them. This same problem accounts for the lower mark of task 7 also. The problem is easily overcome by a simple change of the animation timer allowing the users far more time to traverse the doors.

To find another user who is present in the virtual environment the user simply has to double click on that user's name in the 'Users Logged on' box and they will automatically be beamed to a viewpoint opposite that user's avatar. Not all the users found this feature immediately and this affected the overall average for task 3. Once

this feature was found many of the users thought it was excellent and one chose to comment that in many ways the virtual office is better than a real office due to this ability to jump instantly to a co-worker without having to leave your desk or climb any stairs.

7.4.3 Interactions Within the Private Office Space

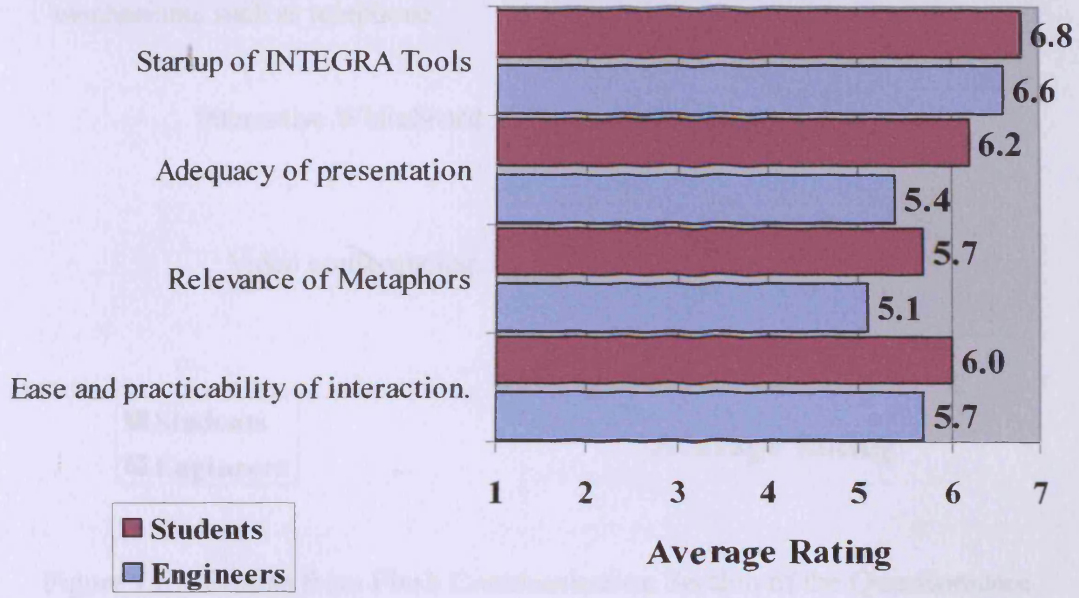

Figure 7.5 – Results from Section 3 of the Questionnaire

Figure 7.5 shows some interesting results, perhaps most surprisingly the relevance of the metaphors. During discussions with the project team, before the evaluation began it was postulated that the engineers would have a better understanding of the metaphors involved than the students. This was not the case as can be seen above. This is perhaps surprising as the VICE system is designed for use by engineers during the conceptual design phase of a project. However, the students grasped the whole concept of a virtual environment better and it is believed that this enabled them to use the system fully intuitively, understanding the metaphors immediately.

All of the users found the start-up of the INTEGRA tools easy. This is not surprising because the metaphors for these tools were obvious although some would argue not true

metaphors. Again the adequacy of the presentation with regards to interactions is sufficient and the users considered the practicality of the interaction to be very good.

7.4.4 Flash Communication Applications

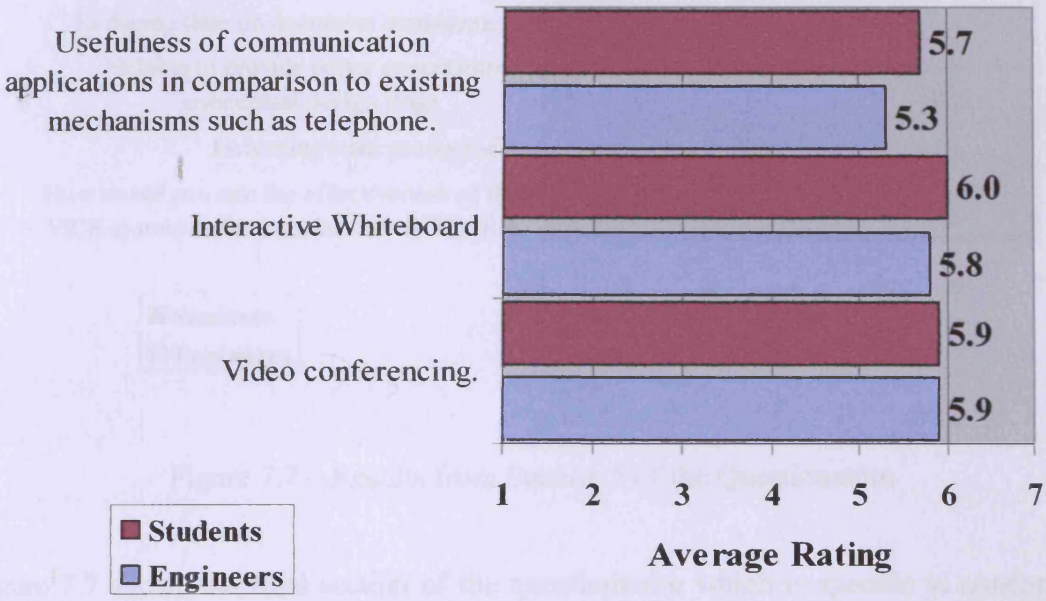

Figure 7.6 – Results from Flash Communication Section of the Questionnaire

Figure 7.6 shows the evaluation of the flash communication tools. These tools are neat and simple to use so it was not a surprise that all of the users arrived at the same positive conclusion. However the engineers were slightly more sceptical about the usefulness of these applications in comparison to existing mechanisms such as the telephone. This is natural because the engineers are generally older, having worked for many years using this older technology. Many of them do not like change and will not fully accept this new technology. Once video conferencing becomes common place it will become a necessity rather than an alternative. Even mobile phone networks have developed the technology so that video conference phone calls can take place over their networks. As the bandwidth gets wider and the cost reduces, communication tools such as video conferencing will become widespread.

7.4.5 Overall Effectiveness of the User Interface

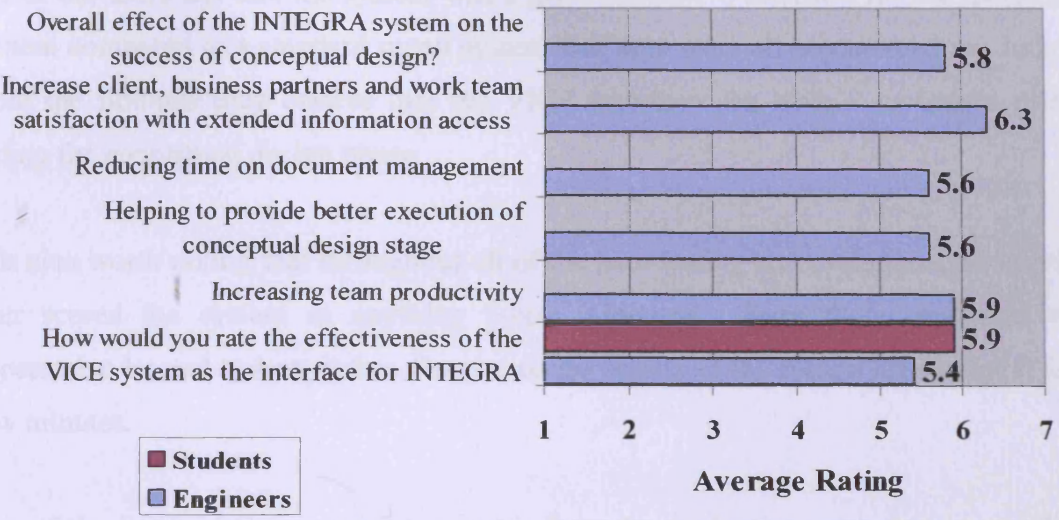

Figure 7.7 – Results from Section 5 of the Questionnaire

Figure 7.7 shows the final section of the questionnaire which is specific to concurrent engineering. Due to the students relative inexperience in working as engineers, they could not be expected to fully grasp what the questions from this section were asking. The questions were graded in relation to what role the VICE interface had to play in it. The highest rated question was concerned work team satisfaction. The engineers believed that the system's ability to improve the ease of access to relevant information was very beneficial. Comments stated that being able to ask another project member a simple question, quickly and easily, even without utilising the flash communication software would relieve a lot of stress and speed the whole process up.

7.5 Conclusions of User Evaluation

All of the users felt that the system was a good alternative interface for the INTEGRA system compared to a standard menu system that they were all accustomed to. Judging from the findings they believe that the VICE interface can have a successful role in aiding the conceptual design phase.

It is also worth noting that throughout all of the user testing and evaluation, no user has ever scored the system as anything below adequate. Even those engineers who appeared to be real technophobes discovered the merits of the system after using it for a few minutes.

One of the few negative comments to arise from the evaluation came from a couple of the engineers. They thought the system was too much like a game and that users would be tempted to use it as such, walking around and wasting time. It is also important to understand that evaluating such a piece of software is difficult in a short space of time. The evaluator will find it difficult to be fully objective because they will have never come across such an interface before. The older generation of engineers may never have experienced this type of navigation altogether, as 3D mouse pointer gaming is relatively new. Ideally the system would be used for an actual design project from start to finish and would lead to a far more accurate evaluation.

The results show that the next generation of engineers in particular will embrace technology and applications such as VICE will become wide spread with collocation becoming unnecessary because of its time consuming and costly nature. Current technology is capable of running VICE at adequate speeds and this will only get better as technology advances.

CHAPTER 8

Conclusions

This concludes the thesis by looking at the successes and failures of the created interface and considering work which could be carried out in the future.

8.1 INTEGRA VICE – Is the Interface a Success?

The aim of this project is to design, develop and test a new style of user interface which promotes a more intuitive form of interaction than the standard desktop metaphor based interface. This new interface has been designed as an alternative for the default interface of the INTEGRA system and must also promote enhanced user collaboration. By choosing alternative metaphors that are more obvious to the user it is postulated that it should be possible for such an interface to be developed.

The user evaluation involving a substantial number of different users proves that the VICE user interface is a successful addition to the INTEGRA system and that it meets the project aims. The user evaluation also provided positive results from two different demographics concluding that the system was easy, intuitive to use and possesses the full functionality that was required.

The use of metaphor based user interfaces is not a new concept. It has become standard practise for most software developers. There are arguments for and against these types of user interfaces. Some advanced users will argue that having such an interface limits their ability to make full use of the applications. However the majority of users do not come within this bracket and for them, metaphor based user interfaces are very useful. This is again evident from the user evaluation.

The thesis examines metaphors and their uses in Information Technology, particularly graphical user interface design. Arguments for and against metaphors exist, however the evidence shows a need for metaphorical based user interfaces rather than interfaces without them. The majority of studies have shown that if metaphors are not present then the user will attempt to use his / her own anyway. The conclusion of the project is that users appear happier to consider using virtual reality and virtual representations (virtual metaphors) rather than the more common user interface metaphors. This VR approach creates a happy medium and eliminates many of the suggested problems with the use of metaphors. The use of a virtual metaphor allows the creation of an interface which is functional yet simple, allowing the users to interact with little or no training. The interface can be used intuitively with the virtual reality creating a metaphor that the users can relate to from their own life experiences. For example, interacting with virtual filing cabinets starts the file management/document handling system BSCW, and face to face meeting of avatars starts video conferencing etc.

There are some existing issues which could be improved. Initially there were plans to incorporate a script that would allow each logged in user to be placed in their own personal office space. This is still possible, however due to time constraints was not achieved before completion of the project. This would be advantageous to the construction project users because they would start in their own personal office and have access to all of the tools without having to traverse unnecessarily. The current interface places each user in the common room upon login. They then have to traverse to a private office space of their choice. This is a slight weakness, but one which could be corrected with further work.

The main achievements of the research project are as follows:

- A thorough analysis of existing collaboration systems and the use of metaphors were completed.
- A virtual office building was created with numerous private offices and communal spaces.
- The virtual environment was distributed across the internet which allowed remote users to access the environment from anywhere with an internet connection.
- Collaboration is achieved using the macromedia flash language which enables fully extensible applications to be written.

- The completed VICE system was successfully integrated into the INTEGRA system.
- A user evaluation was completed which concluded that the completed VICE system was a success and industry members would use it.

8.2 Future Work

For certain areas of the INTEGRA interface it proved difficult to find suitable virtual metaphors. Further work is needed to discover whether there are metaphors which could be used.

There are many other possibilities for user interfaces which could adapt and use the concept of virtual metaphors as an interface. It would be interesting to explore these other areas to find new virtual metaphors so that further collaboration and user interaction could occur.

During the development of the project a major change in the specification of VRML occurred. Because of it's inadequacies, the VRML 97 specification on which the interface was based was scrapped and replaced by an altogether new specification named X3D. The change came too late for this research project and therefore future work would involve changing the interface into an X3D format. X3D is a considerably more mature refined standard than VRML so authors can achieve the behaviours they expect. The web 3D consortium have devised a list of the top 10 reasons why XML based X3D developments are a far superior choice over VRML when creating virtual environments. These reasons can be seen in appendix B. (Web3D Consortium, 2005)

8.3 Future Recommendations

The main recommendation after completion of the project is that it needs to be converted to X3D. This conversion will take some time as there are no autonomous applications that can do it. It is also recommended that more research in to the point of view node or X3D version should be carried out. This node has the capability to greatly reduce traversal times and improve the flow of the virtual environment.

B

Bibliography

About.com. (2004). The History of the Internet. [ONLINE] Available: http://inventors.about.com/library/inventors/blinternet.htm [2004, October 21st]

American Museum of Papermaking. (2004) The Invention of Paper. [ONLINE] Available: http://www.ipst.gatech.edu/amp/collection/museum_invention_paper.htm [2004,October 20th]

Ames, A., Nadeau, D., Moreland, J. (1997). VRML Sourcebook. Vol. 2. John Wiley & Sons Inc, 1997.

Berger, M., Hohl, H., Jarczyk, A., Otto, B., Schneider, M., Volksen, G., (1997). 'CoNus: Workspace-Based Intuitive Collaboration in Virtual Enterprises'. *In: Werner, B. Proceedings of the 6th Workshops on Enabling Technologies*. Cambridge, Massachusetts. IEEE Computer Society. pp. 3 – 9.

Bernstein, M. (1993). Enactment in information farming. *In: ACM Hypertext'93.* New York: ACM Publications, 242-249.

Blau, B., Hughes, C.E., Moshell, J.M. and Lisle, C. (1992). Networked virtual environments. *In: Computer Graphics 1992 Symposium on Interactive 3D Graphics,* p 157.

Blaxxun. (2005). Product Documentation. [ONLINE]. Available: http://developer.blaxxun.com/doc/wwhelp/js/html/frames.htm [2005, August 25th]

Bricken, W. (1990). Virtual Reality: Directions of Growth. *Notes from the SIGGRAPH'90 panel.* [ONLINE]. Available: http://www.hitl.washington.edu/publications/m-90-1/ [2004, December 5th]

Brown, C. Common Front Group. (1995). User Interface Design. [ONLINE]. Available: http://cfg.cit.cornell.edu/cfg/design/contents.html [2003, April 26th]

Building. (1994). Cost model: City of London office blocks, Building, 14 October 1994, pp. 42-48.

Bush, V. (1945). 'As We May Think'. *In: Atlantic Monthly*, Vol. 176(1), pp 101 – 108.

Carlsson, C., & Hagsand, O. (1993). DIVE – a Multi-User Virtual Reality System, *In: IEEE Virtual Reality Annual International Symposium (VRAIS 93)*, pp. 394-400.

Carroll, J.M., Mack, R.L., and Kellogg, W.A. (1988). Interface metaphors and user interface design, *In: Helander, M ed. Handbook of Human-Computer Interaction,* Elsevier Science Publishers B.V. North-Holland, pp. 67-85.

Cen, M., Miles, J., Taylor, M., Bouchlaghem, D., Anumba, C.J., Chim, Y.M. (2002). Requirement Capture for Concurrent Conceptual Design, *In: Schnellenbach-Held, Martina & Denk, Heiko (eds.), Advances in Intelligent Computing in Engineering, Proceedings of the 9th International EG-ICE Workshop*, Darmstadt, pp. 201-211.

Chesher, C. (1994). Colonizing Virtual Reality. Construction of the Discourse of Virtual Reality, 1984-1992. *In Cultronix.* Vol 1. No 1. 1994

Christiansson, P. (2001). Experiences from Using Internet Based Collaboration Tools. 'Konference om Arkitekturforskning og IT'. *Proceedings Conference on Architectural Research and Information Technology. Nordic Association for Architectural Research.* Arkitektskolen i Aarhus 27.-29. april 2001. (pp. 103-112).

Cleetus, K.J., (1992). Definition of Concurrent Engineering. *In: CERC Technical Report Series*, Research Notes, CERC-TR-RN-92-003, Concurrent Engineering Research Center, West Virginia University, Morgantown, pp. 1-5.

Condon, C. (1990). Networked cooperative work: usability issues of MILAN (multimedia industrial local area network). *In: Telematics '90.* BIBA, Bremen, September 1990. pp. 74-101.

Cruz-Neira C., Sandin D.J., DeFanti T.A., Kenyon R.V. and Hart J.C., (1992). The CAVE: Audio Visual Experience Automatic Virtual Environment. *In: Communications of the ACM*, 35 (6): pp65-72.

Cruz-Neira C., Sandin D.J. and DeFanti T.A., (1993). Surround-Screen Projection Based Virtual Reality: The Design and Implementation of the CAVE, *In: Computer Graphics*, 27: pp135-142.

van Dam, A. (May 15, 1996). Graphical User Interfaces: Past, Present and Future. *In: Saul Gorn Memorial Lecture Series Department of Computer Science*, Brown University, Pa

Derbyshire, D., Ferguson, I.A., Muller, J.P., Pischel, M., Wooldridge, M., (1997). Agent-Based Digital Libraries: Driving the Information Economy. *In: Werner, B. Proceedings of the 6th Workshops on Enabling Technologies.* Cambridge, Massachusetts. IEEE Computer Society. pp. 82-86.

Dieberger, A. & Tromp, J. G., (1994). The information city - a metaphor for navigating hypertexts. *In: People and Computers VIII: HCI `94.*, Loughborough, 1994, pp179-194.

Diehl, S. (2001). Distributed Virtual Worlds. Germany: Springer.

DIVE Homepage. (2005) [ONLINE]. Available: http://www.sics.se/dive/ [15th April 2005]

Dorcey, T., (1995). CU-SeeMe Desktop Video Conferencing Software, *In: Connexions 9*

Shaw, C., Green, M., Liang, J., & Sun, Y., (1993). Decoupled Simulation in Virtual Reality with the MR Toolkit. *In: ACM Transactions on Information Systems*, Vol. 11(3)., July 1993. pp 287-317

Shaw, C., Green, M., (1993). The MR Toolkit Peers Package and Experiment. *In: Proc. IEEE Virtual Reality Annual International Symposium*, pp 463-469.

Grimsdale, C., (1991). dVS - Distributed Virtual environment System. *In: Computer Graphics `91 Conference*, London

Dwivedi, S.N., & Sobolewski, M., (1990). Concurrent Engineering - An Introduction. *In: Proceedings of the Fifth International Conference on CAD/CAM Robotics and Factories of the Future '90*, Vol. 1: Concurrent Engineering, New York: Springer-Verlag, 1991., pp 3-16.

Engelbart , D. C., October (1962). Augmenting Human Intellect: A Conceptual Framework. Summary Report, Stanford Research Institute

Engelbart, D. C., & English, W. K., December (1968). A Research Center for Augmenting Human Intellect. *In: AFIPS Conference Proceedings*, Vol. 33, Fall Joint Computer Conference, San Francisco, pp. 395-410

Engelbart, D.C., & Lehtman, H., (1988). 'Working Together' *In: BYTE Journal*, McGraw Hill, Vol. 13, Issue 13, pp. 245 - 252

Evbuomwann, N.F.O., & Anumba, C.J., (1998). 'An Integrated Framework for Concurrent Life-Cycle Design and Construction'. *In: Advances in Software Engineering*, Vol. 29, No. 7-9, Aug-Nov., pp 587-597.

Fisher, D. (1993) 'Communication in Organisations'. West

Florin, F. (1990). Information Landscapes. *In:* Ambron, S. and Hooper, K. (Eds.) *Learning with interactive multimedia.* Redmond, Washington: Microsoft Press, 27-49.

Fraunhoffer FIT. (1995). BSCW. [ONLINE]. Available: http://bscw.gmd.de/index.html [2003, May]

Gillen, D.J. & Fitzgerald, E., (1991) 'Expanding Knowledge and Converging Functions' *In: Concurrent Engineering journal.* May-June 1991, pp.21

Goldenese, B.L., (1993). 'Metrics for Measuring Product Development'. Institute of Tufts University, Medford, Massachusetts

Greenhalgh, C. & Benford, S. (1995). MASSIVE: a Distributed Virtual Reality System Incorporating Spatial Trading. *In: Proc. 15th IEEE International Conference on Distributed Computing Systems (ICDCS '95)*, Vancouver, Canada, May 30-June 2, 1995, IEEE Computer Society, pp 27-34.

Griffin, A., (2004). Where and how was the telephone invented? [ONLINE]. Available: http://www.faqfarm.com/Science/76719 [2004, October 20th]

Griffin, S. (1999). Internet Pioneers. [ONLINE]. Available: http://www.ibiblio.org/pioneers/index.html [2005, January 12th]

Haller, M., Holm R., Volkert, J., & Wagner, R. (1999). VR based safety training in a petroleum refinery. *In: 20th Annual Conference of the European Association for Computer Graphics (Eurographics '99)*, Milano, Italy, September 1999.

Hämmäinen, H. & Condon, C., (1991). Form and room: metaphors for groupware. *In: COCS '91, ACM Conference on Organizational Computer Systems.* Atlanta, Georgia, 5-8 Nov 1991.

Harding, R. Lay, S. Robinson, P. Sheppard, D., & Watts, R. (1997). New Technology for Interactive CAL: The Origami Project. *In: Canole, G. Association for Learning Technology Journal.* University of Wales Press. Vol 5(1).

Haryott, R., Murray, S., Lyall, I., Forster, M., Davis, N., & Jarvis Heckel, I.W.H. (1998a). The Glaxo Wellcome Medicines Research Centre: Concept Design to Detail Design. *In: Proceedings of the Institution of Civil Engineers: Structures and Buildings,* Vol. 128, pp 223 – 233

Haryott, R., Murray, S., Lyall, I., Forster, M., Davis, N., & Jarvis Heckel, I.W.H. (1998b). The Glaxo Wellcome Medicines Research Centre: Detail Design to Construction. *In: Proceedings of the Institution of Civil Engineers: Structures and Buildings*, Vol. 128, pp 234 - 243

Heckel, J., Ganeshan, R., Case, M. & Baskin, A., (1997). 'The Virtual Workspace System (VWS): An Enabling Technology for Collaborating Engineering Applications'. *In: Werner, B. Proceedings of the 6th Workshops on Enabling Technologies.* Cambridge, Massachusetts. IEEE Computer Society. pp. 10 –16.

Henderson Jr, D. A. & Card, S. K. (1986). 'Rooms: the use of multiple virtual workspaces to reduce contention in a window-based graphical user interface'. *In: ACM Transactions on Graphics.* 5 (3) July 1986, pp 211-243.

Hobart, J. (1995). Principles of Good GUI Design. [ONLINE]. Available: http://axp16.iie.org.mx/Monitor/v01n03/ar_ihc.htm [2004, November 29th]

HQpapermaker. (2004) The History of paper. [ONLINE]. Available: http://www.hqpapermaker.com/paper.htm [2004, October 20th]

Hussein, K. Pena-Mora, F., & Sriram, R.D. (1995). CAIRO: A System for Facilitating Communication in a Distributed Collaborative Engineering Environment. *In: Proceedings of Enabling Technologies Infrastructure for Collaborative Enterprise.* Berkeley Spring. WV. IEEE Press. pp.154-162

IBM.com (2004) LDD Today: The History of Notes. [ONLINE]. Available: http://www-10.lotus.com/ldd/whatisnotes [2004, October 21st]

IBM.com[2]. (2004). Ease of Use: Design Basics. [ONLINE]. Available: http://www-3.ibm.com/ibm/easy/eou_ext.nsf/Publish/6 [2004, December 5th]

Karat, C-M., (1998). 'Guaranteeing rights for the user', *In: Communications of the ACM 41*, Vol 12. pp. 29 - 31.

Kobb, S. (2001). Getting Stuck in the MUD. [ONLINE]. Available: http://www.frictionmagazine.com/culturati/gaming/mud.asp [2004, October 21st]

Lakoff, G. & Johnson, M. (1980). Metaphors We Live By. Chicago: Chicago University Press

Landow, G. P., (1992), 'Hypertext and the Convergence of Contemporary Critical Theory and Technology', The Johns Hopkins University Press, Baltimore and London

Loughran, J. & Stahl, M. (1999) The Role of a Distributed Interactive Collaboration Environment (DICE) for Interagency/Military Training, *In: Proceedings of the Command and Control Research and Technology Symposium*,

Love, N., LBL Consulting inc., (1996). 'Better Designs, Faster Development', [ONLINE] Available: http://www.soce.org/papers/lbl_consult/better_faster_dev.htm [2004, December 10th]

Miles, J.M., Anumba, C.J. & Bouchlaghem, N.M., (2001). 'Integration of Multi-Disciplinary Perspectives in Concurrent Conceptual Design'

MOD.uk (2004) "Virtual Reality" Definition. [ONLINE] Available: http://www.mod.uk/issues/simulation/glossary.htm [2004, December 4th]

Nadeau, DR. (1999). 'Advancing 3D through VRML on the Web', *In: IEEE Computer Graphics 19(2)*, pp.4-5

Natale, C., (1993). 'Successful Implementation of Concurrent Engineering Products and Processes' *In: Shina, S.G. ed. Successful Implementation of Concurrent Engineering Products and Processes*. New York: John Wiley & Sons. pp124

National Institute of Standards & Technology, Thomas Group Inc., and Institute for Defence Analyses in Business Week April 30, 1990.

Pejtersen, A. M. & Goodstein L. P., (1988). 'Beyond the Desk Top metaphor: Information Retrieval with an Icon-Based Interface'. *In: Workshop Visualization in Human computer interaction, 7th interdisciplinary workshop informatics & psychology,* Schärding, pp149-182.

Pemberton, L. (1993). Offices, balconies, doors and corridors: an experimental interface metaphor for integrating collaborative design styles. *Lecture notes in computer science: Human Computer Interaction: Vienna Conference, VCHCI '93.* Vol. 733, 419-420.

Prasad, B. (1996). Concurrent Engineering Fundamentals: Integrated Product and Process Organization, p. 1, Prentice Hall, NJ, USA.

Reid, E., (1991). 'Electroplis' [ONLINE]. Available: http://www-courses.cs.uiuc.edu/~cs598kgk/students/srinivasan/srinivasan4.html [2004, February 12th]

Rifas L. (1994). *The dataforest - tree forms as information display graphics.* Dissertation, University of Washington.

Root R. W. (1988). 'Design of a multi-media vehicle for social browsing', *In: SCW 88.* Portland, Oregon, pp 25-38.

Singh, G., Serra, L., Png, W., Wong, A., & Ng, H., (1995). BrickNet: Sharing Object Behaviors on the Net. *In: Proceedings of IEEE VRAIS '95.* pp 44-49.

Sriram, D., Logcher, R., Groleau, and Cherneff, J., (1992). 'DICE: An Object Oriented Programming Environment for Communication, Coordination and Control in Computer Aided Engineering', *In: Artificial Intelligence in Engineering Design*, Edited by Tong, C. and Sriram, D., Academic Press, 1992.

Sriram, R.D. & Logcher, R. (1993). The MIT Dice Project. *In: IEEE Computer. 26(1)*, pp64-65

Stefik, M., Foster, G., Bobrow, D.G., Kahn, K., Lanning, S., Suchman, L., (1987). 'Beyond the chalkboard: computer support for collaboration and problem solving in meetings' *In: Communications of the ACM, Vol. 30 Issue 1.*

Szabó, K. (1995). Metaphors and the user interface. [ONLINE]. Available: http://www.katalinszabo.com/metaphor.htm [2003, April 28]

Taylor, M., Miles, J., Cen, M., Anumba, C.J., Bouchlaghem, D., Yang, H., (2002). Metaphor based interfaces for collaborative concurrent engineering, In: Schnellenbach-Held, Martina & Denk, Heiko (eds.), *Advances in Intelligent Computing in Engineering*, Proceedings of the 9th International EG-ICE Workshop, Darmstadt, pp. 196-200

Taylor, M., Miles, J., Bouchlaghem, D., Anumba, C. J., Cen, M, and Shang, H., (2003), A Metaphor Based 3D Interface for a Concurrent Engineering System, the Nashville Conference, USA.

Taylor, M., Miles, J.C., Bouchlaghem, N.M., Anumba, C.J., Cen, M. & Shang, H., (2004). VRML Virtual Worlds: An alternative to the desktop metaphor for GUI's, in Beucke K. et al (eds) 10th Int. Conf. on Computing in Civil & Building Eng, ISBN 3-86068-213-X, 8pp

Taylor, M., Miles, J.C., Bouchlaghem, N.M., Anumba, C.J. (2005). User Interfaces for Complex Engineering Software: An Application of Virtual Reality. *In ASCE Journal of Computing in Civil Engineering* – subject to alterations

Thalmann, D., Babski, C., Capin, T., Magenat Thalmann, N., & Pandzic, I. S., (1997). Sharing VLNET worlds on the WEB. *In: Computer Networks & ISDN Systems, Vol. 29*, No. 14, pp. 1601-1610

Tuck, M. (2001). The Real History of the GUI. [ONLINE]. Available: http://www.sitepoint.com/article/real-history-gui [2004, Nov 24]

vrmLab. (2005). An Experimental VR Environment. [ONLINE]. Available: http://streamer.rit.edu/~jeffs/vrmLab/ [2005, Sept 02]

Waters, R., Anderson, D., Barrus, J., Brogan, D., Casey, M., McKeown, S., Nitta, T., Sterns, I., & Yerazunis, W., (1997). Diamond Park and Spline: Social Virtual Reality with 3D Animation, Spoken Interaction and Runtime Extendability. *In: Presence, Vol. 6*, No. 4, August 1997, pp. 461-481.

Web3D consortium. (Jan 2005) Why Use X3D over VRML. [ONLINE]. Available: http://www.web3d.org [2005, Sept 03]

Winner, R.I., Pennell, J.P., Bertrand, H.E. & Slusarczuk, M.M.G., (1988). 'The Role of Concurrent Engineering in Weapons System Acquisition', Institute of Defence Analyses Report R-338

Zyda, M.J., Pratt, D.R, Monahan, J.G., & Wilson, K.P., (1992). NPSNET: constructing a 3D virtual world. *In: Computer Graphics, Special Issue on the 1992 Symposium on 3D Interactive Graphics*, 29 March - 1 April 1992, pp. 147-156.

Appendix A : Users Bill of Rights

I. The user is always right. If there is a problem with the use of the system, the system is the problem, not the user.

II. The user has the right to easily install software and hardware systems.

III. The user has the right to a system that performs exactly as promised.

IV. The user has the right to easy-to-use instructions for understanding and utilising a system to achieve desired goals.

V. The user has the right to be in control of the system and to be able to get the system to respond to a request for attention.

VI. The user has the right to a system that provides clear, understandable, and accurate information regarding the task it is performing and the progress toward completion.

VII. The user has the right to be clearly informed about all system requirements for successfully using software or hardware.

VIII. The user has the right to know the limits of the system's capabilities.

IX. The user has the right to communicate with the technology provider and receive a thoughtful and helpful response when raising concerns.

X. The user should be the master of software and hardware technology, not vice-versa. Products should be natural and intuitive to use.

Written by Dr Claire-Marie Karat

Appendix B : Web Consortiums

10 Reasons for using X3D

1. **VRML compatible:** There is still a "Classic VRML" encoding which can play most non-scripted VRML 2 worlds with only minor changes. None of the technology has been lost, but instead it has evolved into X3D. X3D has been designed purposefully to maintain as much compatibility with VRML as possible while still solving incompatibility problems that directly lead to non-interoperability of environments between users.

2. **XML encoding to integrate smoothly with other applications:** XML is fast becoming a prerequisite for including information in corporate and government data bases. Having XML encoding makes it easier to manage, control, validate, and exchange information. The XML encoding of X3D keeps X3D up to speed and allows possible interaction with most information in this world.

3. **X3D scenes and environments operate predictably between different players:** A major problem with VRML is that it is difficult to develop VRML environments that play on all conformant browsers/players. This is because of a lack of adequate specification of VRML behaviour in the VRML standard. The X3D behaviour has been specified far better and in such a way that scenes and environments can interoperate between browsers.

4. **X3D is componentised:** X3D is componentised which means that there is now an allowance for the specification of profiles tailored to a particular large market segment (e.g., CAD, Medical, Visualization). X3D allows cleaner introduction of new technology, which was a severe stumbling block that the industry found with VRML 97.

5. **X3D authoring for any player is consistent and simpler:** The X3D Scene Authoring Interface provides consistent functionality for all scripting languages both internal and external. This is not the case for the VRML 97 specification where Java and ECMAScript have widely different programming models. The X3D SAI solves all of this by specifying a unified set of abstract services that can then be mapped to any programming/scripting language. This in turn enables environments to play consistently regardless of programming language. Language bindings have also been provided for Java and ECMAScript. This makes creating X3D much simpler.

6. **X3D is more feature rich:** A large number of features requested for VRML have been provided in X3D in a manner that is completely integrated into the architecture (as well as being standardised). Thus, many ad hoc solutions that are vendor-specific have been avoided. X3D is basically "VRML3" with the problems and issues communicated from the users about VRML 97 fixed in X3D.

7. **X3D is continually being enhanced and updated:** X3D is growing in functionality. The Proposed Draft Amendment 1 specification adds such things as 3D textures and shading languages is available. This also corrects some identified anomalies in the original X3D specification. The structure of X3D makes it much easier to update on a regular basis. It is also easier to add new features that adapt to the changing graphics and commercial markets.

8. **X3D applications can be certified as reliable and predictable:** An X3D conformance program is being developed by the Web3D Consortium to provide service marks for conformant X3D software. This will ensure that authoring and playback applications (browsers/players) will be reliable and predictable.

9. **An X3D open source conformant application is available as a developer resource**: An open source implementation of nearly all of X3D (Xj3D) is available and proprietary conformant browsers such as Flux are also being

developed. Unlike with VRML scenes, X3D scenes will play consistently in each conformance certified player.

10.X3D binary format offers encryption (i.e. security) and compression (i.e. speed): A Compressed Binary encoding is under development that allows encryption for model security and very high compression of X3D environments. Scene parsing and loading speedups of 300-500% are commonplace. It will also be easy for all browsers to support all encodings because the only difference with them will be a slightly different parser required. Current X3D browser developers plan to support all of the encodings.

. (Web3D Consortium, 2005)

Appendix C :

Example Questionnaires from User Evaluation

INTEGRA VICE Evaluation Questionnaire

After using the INTEGRA VICE software how would you rate the following aspects of the system in order for the tool to be effectively used?

Please fill out your details and then answer questions, selecting values (1: very poor to 7: excellent) or n/a (not applicable) for the criteria listed below.

NAME: NICK PHILLIPS

OCCUPATION: PRODUCTION MANAGER

DESCRIPITON OF WORK: ORGANISATION + MANAGEMENT OF STAFF IN A LIGHT MANUFACTURING ENVIRONMENT.

COMPUTER USAGE

- where? HOME + WORK.
- what for? GAMES + WEB BROWSING + OFFICE APPLICATIONS. + PROGRAMMING.
- connection speed? BROADBAND (568K)

The Virtual Office Building	Very Poor	Poor	Less than adequate	Adequate	Good	Very Good	Excellent	Not Applicable
• Ability to move around the virtual environment.	1	2	3	(4)	5	6	7	n/a
• Interaction with other users.	1	2	3	4	(5)	6	7	n/a
o Gestures	1	2	3	4	(5)	6	7	n/a
o Chat	1	2	3	4	5	(6)	7	n/a
• Adequacy of presentation.	1	2	3	4	5	6	(7)	n/a

Any other comments:

Completion of simple tasks Please grade against the ease with which you completed the task	Very Poor	Poor	Less than adequate	Adequate	Good	Very Good	Excellent	Not Applicable
• Task 1: Register with the VICE system	1	2	3	4	5	6	(7)	n/a
• Task 2: Traverse to another floor within the office building	1	2	3	4	5	6	(7)	n/a
• Task 3: Find another user	1	2	3	4	5	6	(7)	n/a
• Task 4: Traverse to a private office place	1	2	3	4	(5)	6	7	n/a
• Task 5: Invite someone to your office through text chat	1	2	3	4	5	6	(7)	n/a
• Task 6: Start a video conference and/or whiteboard application with co-worker	1	2	3	4	5	6	7	(n/a)
• Task 7: Traverse to a boardroom	1	2	3	4	(5)	6	7	n/a
• Task 8: Start a boardroom conference	1	2	3	4	5	6	(7)	n/a

Any comments about any tasks:

Interactions within the private office space	Very Poor	Poor	Less than adequate	Adequate	Good	Very Good	Excellent	Not Applicable
• Ease and practicability of interaction.	1	2	3	4	(5)	6	7	n/a
• Relevance of Metaphors	1	2	3	4	5	(6)	7	n/a

Example: The filing cabinet metaphor (a virtual filing cabinet)
Please comment:

HOW ABOUT A JUKEBOX FOR PLAYING MUSIC?
TV FOR PLAYING VIDEOS?
VIDEO GAME FOR PLAYING GAMES?
PROJECTOR FOR PRESENTATIONS?

• Adequacy of presentation	1	2	3	4	5	(6)	7	n/a
• Startup of INTEGRA Tools	1	2	3	4	5	6	(7)	n/a

Any other comments:

Flash Communication applications

• Video conferencing.	1	2	3	4	5	6	7	(n/a)
• Interactive Whiteboard	1	2	3	4	5	6	7	(n/a)
• Usefulness of communication applications in comparison to existing mechanisms such as telephone.	1	2	3	4	5	(6)	7	n/a

Any other comments: AN INTERACTIVE WHITEBOARD COULD BE EXTREMELY USEFUL FOR REMOTE PRESENTATIONS

Overall effectiveness of the user interface	Very Poor	Poor	Less than adequate	Adequate	Good	Very Good	Excellent	Not Applicable
How would you rate the effectiveness of the VICE system as the interface for INTEGRA in terms of its application integration, in contrast to a menu type interface?	1	2	3	(4)	5	6	7	n/a
What role does the VICE User Interface play in	1	2	3	4	5	6	7	n/a
• Increasing team productivity	1	2	3	4	5	6	7	n/a
• Helping to provide better execution of conceptual design stage	1	2	3	4	5	6	7	n/a
• Reducing time on document management	1	2	3	4	5	6	7	n/a
• Increase client, business partners and work team satisfaction with extended information access	1	2	3	4	5	6	7	n/a
Any other comments: VICE IS CURRENTLY QUITE SLOW TO MOVE AROUND THE OFFICE. A SIDE-STEP FUNCTION WOULD INCREASE THIS DRAMATICALLY ALSO, CRASH DETECTION (OR C)								
How would you rate the overall effect of the INTEGRA system on the success of the conceptual design stage?	1	2	3	4	5	6	(7)	n/a
If you could do one thing to improve the INTEGRA VICE interface, what would it be?								
Please write answer here: SEE ABOVE - CRASH DETECTION + SIDE-STEPPING USE SOME IMPROVE								

INTEGRA VICE Evaluation Questionnaire

After using the INTEGRA VICE software how would you rate the following aspects of the system in order for the tool to be effectively used?

Please fill out your details and then answer questions, selecting values (1: very poor to 7: excellent) or n/a (not applicable) for the criteria listed below.

NAME: Jonathan Matthew

OCCUPATION: Scientist.

DESCRIPITON OF WORK: ossice work + some lab work. Work from home sometimes.

COMPUTER USAGE

- where?
- what for?
- connection speed?

The Virtual Office Building	Very Poor	Poor	Less than adequate	Adequate	Good	Very Good	Excellent	Not Applicable
1. • Ability to move around the virtual environment.	1	2	3	4	(5)	6	7	n/a
2. • Interaction with other users.	1	2	3	4	5	(6)	7	n/a
3. o Gestures	1	2	3	(4)	5	6	7	n/a
4. o Chat	1	2	3	4	5	(6)	7	n/a
5. • Adequacy of presentation.	1	2	3	4	(5)	6	7	n/a

Any other comments:

1. Potentially better than an ossice, because double click gets you where you want to be instantly* Very good for disabled people who can't get around. Virtual building needs to be a reconstruction of your actual ossice ~~else~~ to make it more conceptual.
3. It would be better if ~~their~~ peoples actual faces were real time mapped imposed on the virtual characters so you can see if they look to busy to speak to etc.

* makes working more efficient.

Completion of simple tasks Please grade against the ease with which you completed the task	Very Poor	Poor	Less than adequate	Adequate	Good	Very Good	Excellent	Not Applicable
• Task 1: Register with the VICE system	1	2	3	4	5	6	(7)	n/a
• Task 2: Traverse to another floor within the office building	1	2	3	4	5	6	(7)	n/a
• Task 3: Find another user	1	2	3	4	5	6	(7)	n/a
• Task 4: Traverse to a private office place	1	2	3	4	5	(6)	(7)	n/a
• Task 5: Invite someone to your office through text chat	1	2	3	4	5	6	(7)	n/a
• Task 6: Start a video conference and/or whiteboard application with co-worker	1	2	3	4	5	6	(7)	n/a
• Task 7: Traverse to a boardroom	1	2	3	4	5	6	(7)	n/a
• Task 8: Start a boardroom conference	1	2	3	4	5	6	(7)	n/a

Any comments about any tasks:

Interactions within the private office space	Very Poor	Poor	Less than adequate	Adequate	Good	Very Good	Excellent	Not Applicable
• Ease and practicability of interaction.	1	2	3	4	(5)	6	7	n/a
• Relevance of Metaphors	1	2	3	(4)	5	6	7	n/a
Example: The filing cabinet metaphor (a virtual filing cabinet) Please comment:								
• Adequacy of presentation	1	2	3	4	(5)	6	7	n/a
• Startup of INTEGRA Tools	1	2	3	4	5	(6)	7	n/a
Any other comments:								
Flash Communication applications								
• Video conferencing.	1	2	3	(4)	(5)	6	7	n/a
• Interactive Whiteboard	1	2	3	(4)	(5)	6	7	n/a
• Usefulness of communication applications in comparison to existing mechanisms such as telephone.	1	2	3	(4)	(5)	6	7	n/a
Any other comments:								

Overall effectiveness of the user interface	Very Poor	Poor	Less than adequate	Adequate	Good	Very Good	Excellent	Not Applicable
How would you rate the effectiveness of the VICE system as the interface for INTEGRA in terms of its application integration, in contrast to a menu type interface?	1	2	3	4	5	6	(7)	n/a
What role does the VICE User Interface play in	1	2	3	4	5	6	7	n/a
• Increasing team productivity	1	2	3	4	5	(6)	7	n/a
• Helping to provide better execution of conceptual design stage	1	2	3	4	(5)	6	7	n/a
• Reducing time on document management	1	2	3	4	5	(6)	7	n/a
• Increase client, business partners and work team satisfaction with extended information access	1	2	3	4	5	6	(7)	n/a
Any other comments:								
How would you rate the overall effect of the INTEGRA system on the success of the conceptual design stage?	1	2	3	4	5	(6)	7	n/a
If you could do one thing to improve the INTEGRA VICE interface, what would it be?								
Please write answer here: