Department of Architecture & Building Science

University of Strathclyde,

Glasgow, Scotland,

United Kingdom

Communication and Collaboration within a VR System for Architectural Design

GIULIANA UCELLI

A thesis presented in fulfilment

of the requirements for the degree of

DOCTOR OF PHILOSOPHY

October 2002

THESIS CONTAINS CD ROM

Acknowledgments

I would like to express my gratitude to the people that contributed to the completion of this work and to the development and testing of JCAD-VR. Prof. T.W. Maver for his support, supervision and trust during the course of the research and for giving me the opportunity to work at ABACUS. Malcolm Lindsay for his advice, practical help and friendship during these three years. Dr. Jelena Petric for suggestions, ideas and practical help especially during the experiment. Dr. Mike Grant for discussions and technical support. Students Christoph Ackermann, Ross Marshall and Edward Wright for taking an active part during the in-house testing of the software.

I would like to thank Dr. H. Achten, Prof. B. de Vries and A. J. Jessurun at the Design System Group, TU/e Eindhoven, for feedback and suggestions about JCAD-VR and for their patience in the initial stages of the cross-university experiment.

Thanks also to Prof. J. af Klercker for the initial support and trust and to Dr. Wolfgang Dokonal for his enthusiastic feedback about JCAD-VR.

My gratitude goes to my colleague and friend Jasna Petric for the happy and sad moments spent together and for her support, to Jose Ripper Kos for bringing Brazilian humour from time to time and for the invitation to his university in Rio and to Dr. Massimo Calligaris who "animated" most of my stay in Glasgow.

I would like to thank also Alessandro, Alain, Alberto, Giampaolo, Giselle, Kalliopi, Marcelo, Niall, Ombretta, Pamela, Paolo, Roberto, Tanja and Thomas for the nice time spent together.

Special gratitude goes to my parents for their love and especially to my mother who makes me feel supported even from thousands of kilometres away. To Gaetano Corleo for the initial psychological and financial support. To my uncle Bruno for the interesting discussions and valuable suggestions. To Gianluca, Gabriele, Mariangela and Rossella for their love. Last but not least, to Giuseppe who helped me with love and patience in these last years.

Abstract

This thesis addresses issues related to the development of Collaborative Virtual Environments (CVEs) for architectural design use. Today a new level of complexity has been brought to the meaning of virtuality by the creation of network-based virtual communities and the use of avatars along with multimedia which provides the technology for remote presence and collaborative experience. Communication and especially collaboration among design teams are now key factors in making the design process faster and more efficient in order to achieve increased competitiveness in the construction market.

The objective of this thesis is to present a tool that is capable of creating 3D shapes in a shared VR environment, therefore allowing the evolution of the design to be a shared process. Along with its companion thesis (Conti, 2002) it gives the description of a framework and software prototype which could help practitioners using Virtual Reality technology by being a new interface for collaborative design at the early stages of the design process. The prototype system that is described here is called Java™ Collaborative Architectural Design tool in Virtual Reality (JCAD-VR) and this thesis presents a description of its collaborative architecture.

This thesis gives a description of the phases and technical solutions in the development of the network architecture and collaborative features of the JCAD-VR system. In addition, several different communication tools were used to enhance communication and the flow of information among the design teams and they are integrated in a Virtual Environment specifically created for architects.

After the description of the software development the results are given of a test of the collaborative architecture of JCAD-VR and its ease of use in a real multi participant design experiment which show the potential and efficiency of using CVEs in architecture.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AEC	Architecture Engineering Construction
API	Application Programming Interface
CAAD	Computer Aided Architectural Design
CAD	Computer Aided Design
CAVE	Cave Automated Virtual Environment
CHI	Computer Human Interaction
CMC	Computer Mediated Communication
CSCW	Computer Supported Cooperative Work
CVDE	Collaborative Virtual Design Environment
CVE	. Collaborative Virtual Environment
DARPA	Defence Advanced Research Projects Agency
DIS	Distributed Interactive Simulation
f2f	Face to Face
GDSS	Group Decision Support Systems
HCI	Human-Computer Interaction
HMD	Head-Mounted Display
HUD	Heads-Up Display
I/O	Input/Output
IT	Information Technology
JCAD-VR	Java™ Collaborative Architectural Design tool in Virtual Reality
JMF™	Java™ Media Framework
JVM™	Java™ Virtual Machine
MOO	Multi-user Object Oriented system

MUD	Multi-User Dimension
OA	Office Automation
OOP	Object-Oriented Programming
RTP	Real-Time Transport Protocol
SE	Software Engineering
TCP/IP	Transmission Control Protocol/Internet Protocol
VE	Virtual Environment
VR	Virtual Reality
VRAD	Virtual Reality Aided Design
VRML	Virtual Reality Modelling Language
WWW	Worlds Wide Web

1 Introduction

1.1 Introduction

This chapter introduces the general aim of the thesis, its general framework and the most important concepts at the foundation of the project.

The research outlined in this thesis addresses the issues concerning the support of the architectural design process by computer applications. In particular this project investigates the use of software that supports architects at the early stages of the design process, at the so-called conceptual modelling phase. Regarding conceptual design, Shukur (2000) stated, that designing a building or other artefact involves really great effort to produce an original concept. In fact due to its nature, concept modelling is to a certain extent more complex than the later design stages since it has less reference to reality. This can potentially cause many aspects and features of its design to be lost or inadequately handled depending on the subjects' capabilities (Shukur, 2000).

Traditionally many architects have addressed the problem of abstraction that is typical of conceptual modeling by proving their design proposals using physical models. Unfortunately the most accurate 2D paper representations are not usually suitable to explain and transmit the complexity of some architectonic ideas.

Nowadays advances in computer hardware and software have significantly contributed to this issue allowing 3D virtual modelling and animations to be used daily in the architectural practice. Similarly the relatively recent introduction of Virtual Reality (VR) has represented another step towards a fully comprehensive simulation technology. The next section will discuss the introduction and application of VR to the architectural design process.

1.1.1 Virtual Reality as a New Means for Designing Architecture

Although today the use of the *third dimension* has become a daily practice, the Computer Aided Architectural Design (CAAD) *community* is only now

experiencing the move from static representation, based on 2D renderings or pre-recorded animations, to dynamically generated 3D representations. Real-time navigation and interaction, typical of Virtual Reality environments, provide the fluent interface and that facilitates the exploration of the design proposal, that is the main omission of all the Computer Aided Design (CAD) packages commonly in use.

Furthermore the access to desktop VR applications makes them a feasible approach in everyday practice thanks to the increasing growth of computational resources and hardware power. Moreover, the recent growth of network-based virtual communities and the use of avatars have brought a new level of complexity to the meaning of *virtuality*, providing the technology for remote presence and collaborative experiences.

The use of VR in design broadens the boundaries of traditional perception by providing experiences of worlds not necessarily real or material. Therefore it gives the user the freedom to simulate and eventually to build up knowledge and skills dangerous or too expensive for human beings to acquire. In an architectural context the use of Virtual Reality provides the designer with an appropriate, quick and practical feedback that facilitates the search for design solutions. In fact, due to its visualisation power, it enables the capturing of more information than would be possible with the use of the traditional media and it makes the checking of the design solutions more efficient by enhancing simulation capabilities. Furthermore VR provides a natural and user-friendly interface between practitioners and clients enabling them to check the functionality of the design and ensuring that the design meets the clients' expectations. Consequently VR could become the ideal simulation medium for architects investigating design solutions and it could contribute to the production of a better-built environment by addressing sustainability through environmental simulations and appraisal, and engaging design creativity through immersive design.

Thanks to all these advantages it is highly predictable that in the near future VR will become the interface for the next generation of CAD applications, the so called VRAD (Virtual Reality Aided Design) systems thus promoting VR from a mere presentation medium to a more powerful and effective design tool. Moreover

current research interest is in multidisciplinary working activities and collaborative networks which broaden the concept of VR itself through the development of multi-user applications that allow several remote participants to interact within the virtual environment and to accomplish, collaboratively, complex design tasks.

1.2 The Aim of the Project

The importance of VR technology with its revolutionary shift in visualization and the effect that VR-based applications could have for the architectural design process is stressed by Dagit (1993, p.514) who states that "architects, as a group, are more aware than most of the profound impact that Brunelleschi's invention of perspective had on society following the fifteenth century. Perspective initiated a fundamental change in the way humans perceive themselves and their environment. Some are looking to virtual worlds as a similar key to opening up new levels of human perception."

In particular, the identification of the role of VR and its place within the practice of architecture has been the subject of a previous experience of the author published at the UK VR-SIG conference in Glasgow (Ucelli et al., 2000). During the first year of the PhD course the author had the chance to work part-time at the Glasgow office of the internationally renowned engineering consultants company Ove Arup & Partners. This office was particularly interested in coming closer to the new visualization and simulation capabilities allowed by Virtual Reality technology. The time spent working at Arup provided the opportunity to investigate problems and issues about the use of 3D modeling and VR in ongoing projects. This experience gave the chance to work closely with both architects and engineers who were for the first time experiencing VR technology. During a 6 months time a number of 3D models were created and architects and engineers were invited repeatedly to evaluate and discuss the progress of the projects at the Virtual Environment Laboratory (VEL) in the ABACUS unit of the University of Strathclyde, in Glasgow (University of Strathclyde, 2002).

This experience confirmed what other authors had stated (Dorta et al., 1998, p. 144): "the design process is made up of two activities which are graphic ideation

or conceptual design and communication [...] the first is a formative process dealing with creating and evolving ideas [active phase]; the second is a descriptive process aiming at presenting to others fully-formed ideas [passive phase]".

Unfortunately the results of the experience were such that, while proving the huge potential of VR for reducing designs "flaws", they highlighted that the present use of VR is limited only to the last phase of the design process, which Dorta et al. (1998) call the passive phase, when the visualization takes place and the result of the design process is eventually shown to the client. Figure 1.1 resumes the most important phases regarding the development of CAAD and VR models. The schema is the result of direct observation of project dynamics in the real working environment experienced by the author.

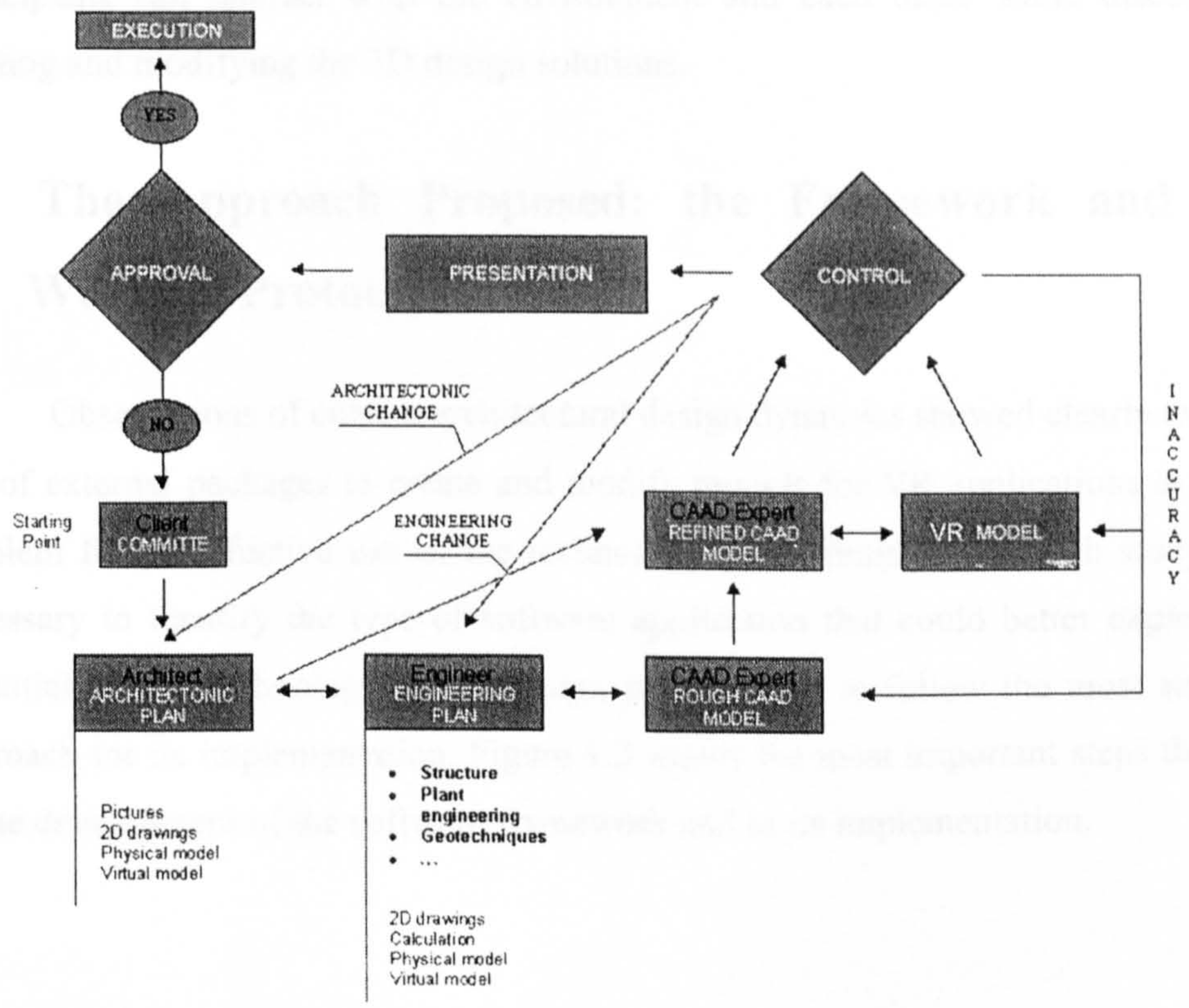


Figure 1.1: The architectural design process from a 3D modelling/VR point of view

It is evident from the graph in Figure 1.1 that the use of VR-based visualization techniques is confined to the end of the conceptual design phase after all the design choices have been taken. This is due mainly to the fact that the models

to be visualized in VR have been created using traditional CAD/CAAD packages and not directly within the virtual environment, making the use of VR more time consuming and therefore less effective.

The knowledge gained through this valuable experience was the base of the concept of the framework reported in this thesis.

The concept upon which the Java™ Collaborative Architectural Design tool in Virtual Reality (JCAD-VR) framework is founded is to try to anticipate the use of VR within the active phase of the design process thus taking full advantage of VR technology and exploiting its creative potential. The aim is to provide the designer with a tool for creating 3D-shapes in a shared VR environment, thus allowing the design to be shared as it evolves. VR then becomes a new design tool and multi-participants can interact with the environment and each other while discussing, creating and modifying the 3D design solutions.

1.3 The Approach Proposed: the Framework and the Working Prototype

Observations of current architectural design dynamics showed clearly that the use of external packages to create and modify models for VR applications is a key problem for an effective use of the technology. A preliminary research study was necessary to identify the type of software application that could better exploit the potential of VR technology in the design process, and to follow the most suitable approach for its implementation. Figure 1.2 shows the most important steps that led to the development of the software framework and to its implementation.

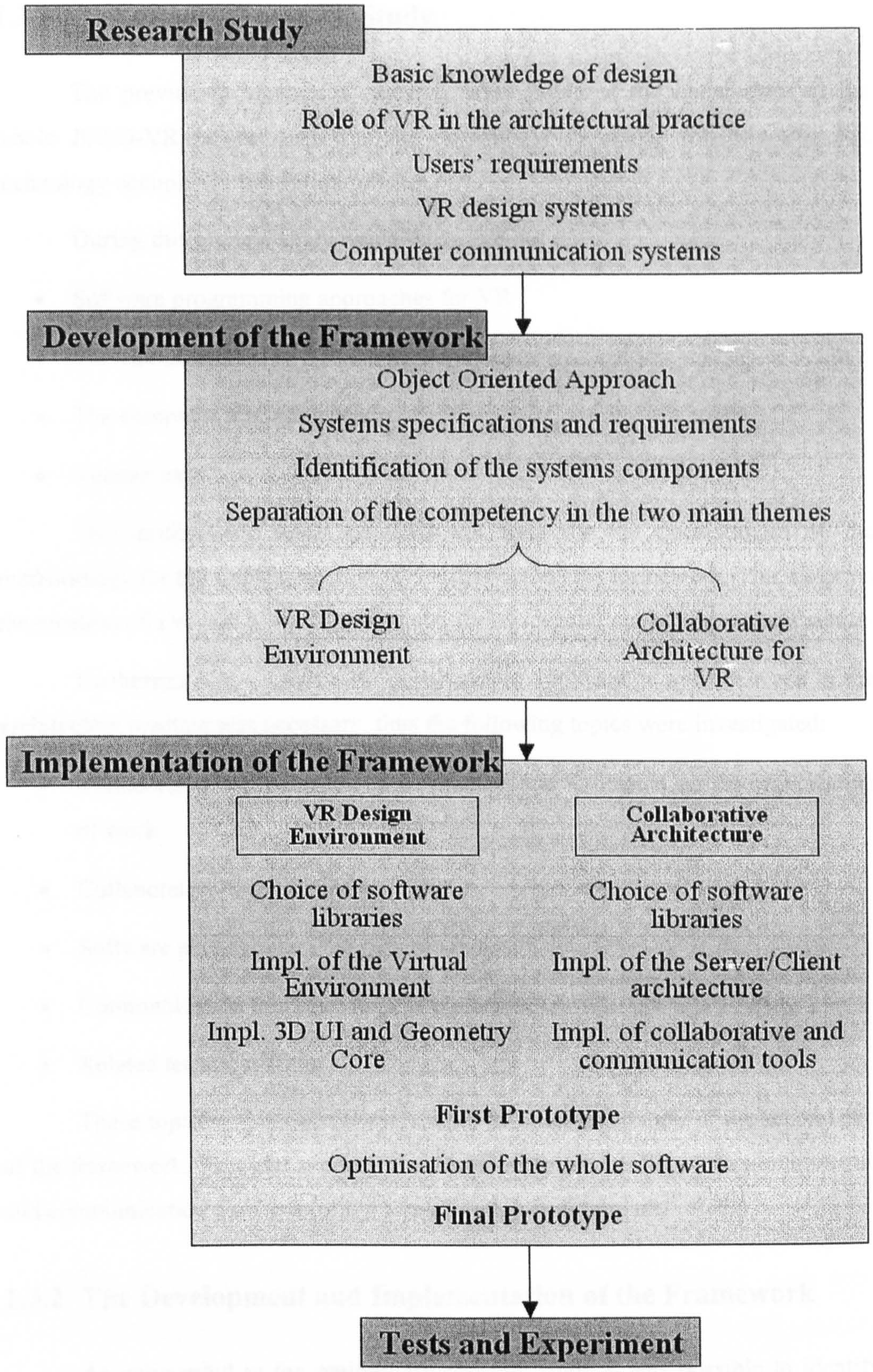


Figure 1.2: Main phases of the project

1.3.1 Preliminary Research Study

The previously mentioned research study provided the foundations of the whole JCAD-VR project and was the opportunity to clarify the role that VR technology occupies in the design process.

During this phase a study was also carried out on:

- Software programming approaches for VR
- The characteristics of VR software packages
- The computer platforms used
- Related technical issues.

This preliminary study provided the base for the development of the methodology for the implementation of the first part of the framework. This supports the creation of a virtual design environment for conceptual modelling in architecture.

Furthermore an overview on collaborative applications and their use in the architecture practice was necessary, thus the following topics were investigated:

- Collaboration in the architectural practice and its impact on the organization of work
- Collaborative software architectures
- Software packages and their characteristics
- Communication tools and their implementation
- Related technical issues.

These topics address issues relevant to the implementation of the second part of the framework. This part supports the development of a collaborative architecture and communication platform for the virtual design environment.

1.3.2 The Development and Implementation of the Framework

As anticipated in the preliminary research study it was possible to identify two independent, but related, research themes within the JCAD-VR project:

1. The creation of a Virtual Design Environment
2. The implementation of its collaborative architecture.

Each theme required addressing a multitude of theoretical and technical issues and a great deal of attention was paid to the creation of a coherent unique application using independent compatible software modules. Figure 1.3 shows the five main sectors that form the framework of the project: 3D Interface, 3D Geometry Core, Computer Supported Cooperative Work (CSCW) tools, Collaborative Architecture and Database Management. Those sectors marked in red refer to the development of the Virtual Design Environment, those in yellow to the implementation of Collaborative Architecture, including the necessary communication tools and a module for the management of shared databases.

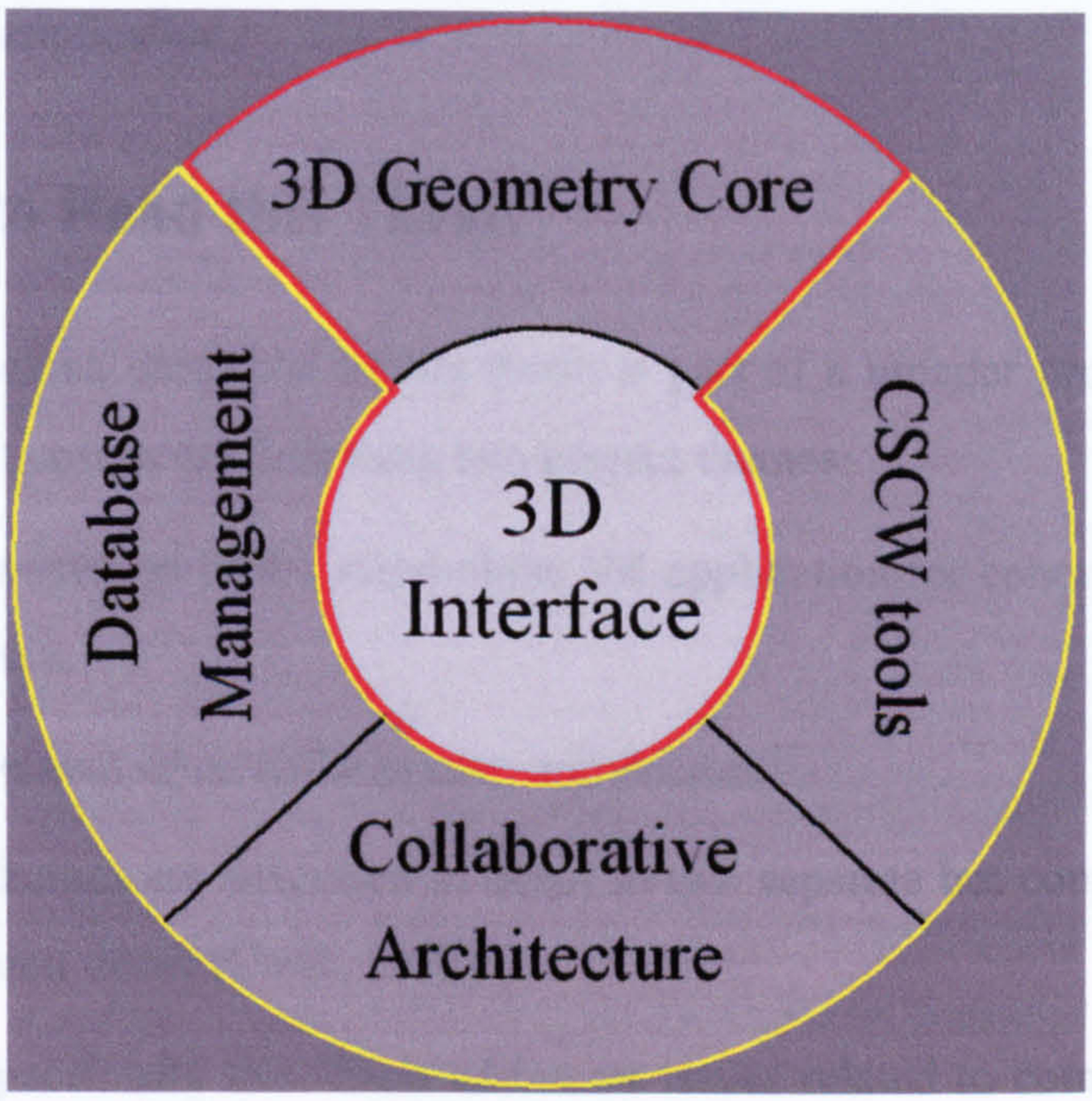


Figure 1.3: Overview of the whole software framework with the two parts marked in yellow and red

Once the general framework of the project was established, each area of interest was independently developed, although every sector was checked against the general framework to ensure compatibility.

The implementation of the two sectors proceeded in parallel as shown in Figure 1.2 until the completion of the first fully working prototype. Then after further software optimisations, the final prototype was developed.

1.3.3 Tests and Experiments

Many unreported tests were carried out during the implementation of the prototype in order to check for software mistakes and bugs in the code. Since the entire project focuses on the early stages of the architectural design process the final prototype was tested in a real working situation involving the creation of an initial conceptual model for a design project.

The experiment that involved students of architecture will be discussed in detail in Chapter 7. It gave the opportunity to test the stability, ease of use and efficacy of the application.

1.4 How to Read this Thesis

The research described in this thesis is part of a broader project outlined in Section 1.3 that covers the following two project themes:

1. The implementation of the stand-alone VR application for conceptual modelling in architecture.
2. The development of its collaborative architecture.

These themes are described in detail in two separate but coupled theses, this and its companion thesis (Conti, 2002).

More specifically this thesis addresses issues related to communication and collaboration activities in architectural practice and it outlines the most important phases of the development of the collaborative architecture of JCAD-VR. The companion thesis (Conti, 2002) highlights theory on user-interfaces in VR environments and it describes the methodology followed to implement the stand-alone Virtual Reality application, called JCAD-VR, specifically designed for the early stages of architectural design.

Each thesis can be read separately, since each one covers different aspects of Virtual Reality applications, or read together as part of a broader project. It is nevertheless highly recommendable to read them together in order to gain a comprehensive view of the whole project.

Both theses are structured symmetrically (See Figure 1.4), with four shared chapters.

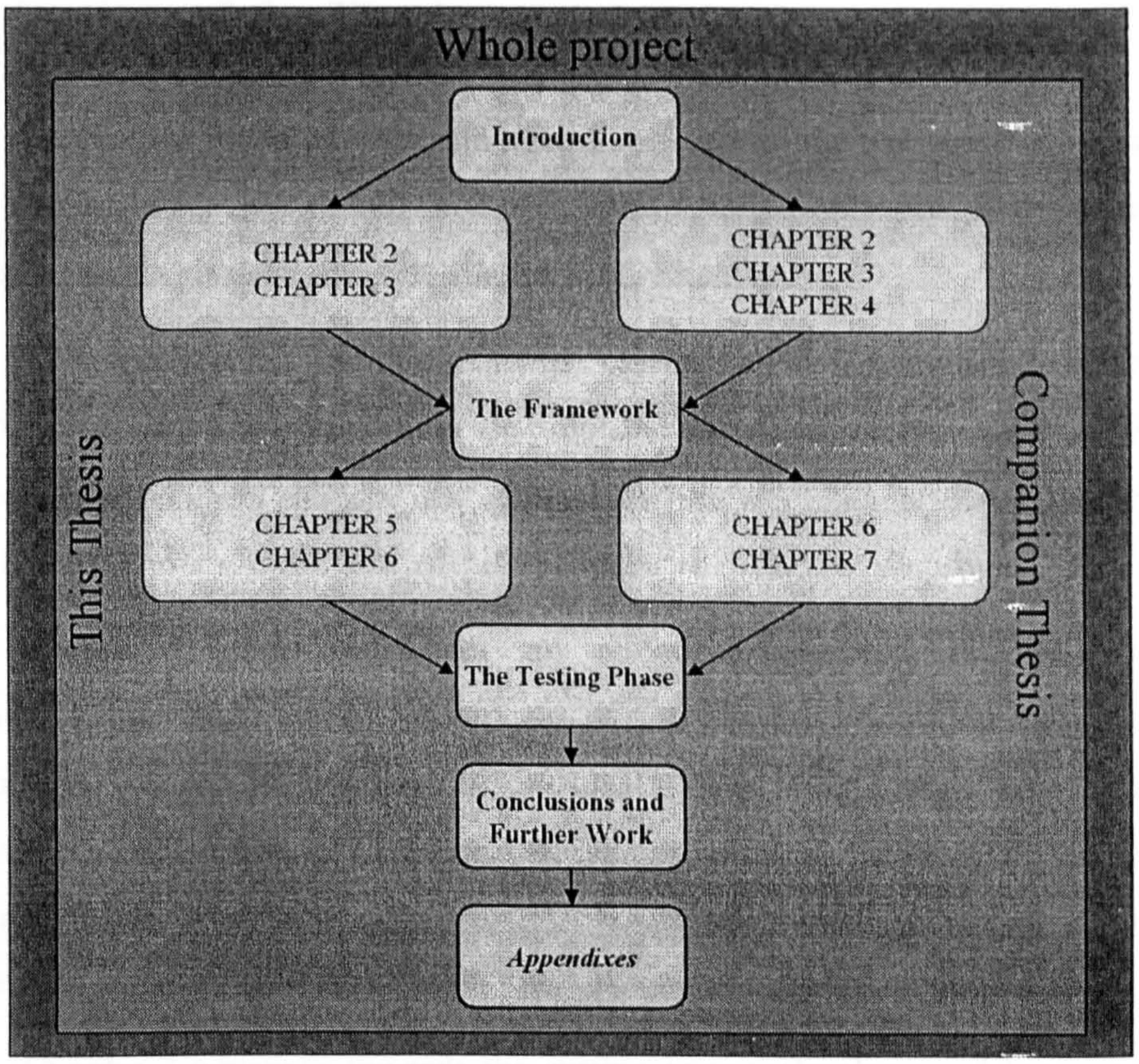


Figure 1.4: The symmetrical structure of the two theses with the shared chapters placed at the centre of the structure

The Chapters number *one*, *four, seven* and *eight* are common and therefore they are included in both theses, since they refer to the general topics that were shared by the two theses. Specifically:

- *Chapter One* provides the overall introduction to the project
- *Chapter Four* describes the general structure of the framework and of the JCAD-VR prototype

- *Chapter Seven* highlights the exercises set up for the testing phase of the prototype
- *Chapter Eight* draws the conclusions and proposes further developments.

However, Chapters number *two*, *three*, *five* and *six* are specific to this thesis and they describe in detail issues and methodologies relevant to the individual work.

1.5 Outline of the Thesis

Section 1.4 clarified the symmetrical structure of the two theses covering the whole project. This section will provide a more detailed overview of the content of this particular thesis describing its chapters individually.

This outline to the thesis will present first a general overview of Collaborative issues in VE and then the proposed working prototype: JCAD-VR. The common Chapter 1 introduced the general overview of the research themes and clarified how to view this thesis within the scenario of the whole project.

Chapter 2 then introduces an historical overview on the role of communication within the design process. It outlines the evolution of theories and technologies related to communication issues within the design process and it proposes a classification of Collaborative Design solutions based on different types of communication systems. This chapter addresses issues related to Computer Supported Cooperative Work (CSCW) and general collaborative applications.

Chapter 3 advances towards the cutting edge-technology and focuses on the use of VR as the ultimate medium for Collaborative Design systems. It introduces the basic concepts of a Collaborative Virtual Environment (CVE) and it offers a comprehensive classification of existing CVEs according to their purpose and technical solution.

The common Chapter 4 is the start of the second, more technical part of the thesis, and it introduces an overview of the methodology followed to develop the project framework and the JCAD-VR prototype. It describes general technical choices, it presents the client-server architecture of the prototype and it introduces

the Collaborative-based unit which is described in depth in the following two chapters.

Chapter 5 provides a structured overview of the Collaborative-based unit: it introduces the methodology followed to implement the collaborative architecture of JCAD-VR, it justifies the choice of the programming language and it presents the adopted software libraries. This chapter describes in detail the server unit and it presents network solutions that are included in the framework, providing the reader with a comprehensive view of the package and its modules.

Chapter 6 describes in detail the implementation and technical issues of the client network unit. All the relevant classes and software solutions are highlighted and justified in this chapter. The client unit, with its sub sections addressing network transfer and database handling issues, are described in depth. Technical solutions related to the implementation of communication channels such as video-conferencing facilities will be fully described.

The common Chapter 7 reports on the preparation and the execution of an experiment in the actual use of the JCAD-VR prototype. The experiment offered the opportunity to test the functionality of the software while several users were accomplishing collaboratively a given design task.

Finally Chapter 8 concludes the thesis with the results acquired during the simulation. It points out the research achievements and contributions, and it highlights further developments.

1.6 Conclusions

This first chapter gave a general introduction to the aspects to be covered in the rest of the thesis. It highlighted the structure of the JCAD-VR project especially, showing how this was developed by the combined effort of our research.

As shown the coupled theses have some common chapters in order to make them easier to read. The introductory and concluding chapters are common ones, while the other chapters will address only the research literature relevant to their

aspects of the project, specific issues and different parts of the implementation of the JCAD-VR system.

Chapter 2 will address more specific issues relative to communication and collaboration as the means to enhance the design and decision-making process.

2 An historical overview on the role of communication within the design process

2.1 Introduction

This chapter, and Chapter 3, will form the foundation of this research and they will present the theoretical background to the development of the JCAD-VR system. Communication and collaboration will be highlighted according to different perspectives: communication as a medium and communication and collaboration as a means to enhance the decision-making and design processes. The revolution in communication, the technology behind it with its impact on society and in work practice, and collaboration as a new approach to design activity will be also discussed. In particular, Section 2 will discuss the act of designing as a communication activity and communication in relation to the practice of architecture. Section 3 will present the phenomena of the communication era, the enhancement of telecommunication technology and its impact on modern working practises. Section 4 will show the scientific disciplines that are the theoretical background for collaboration and cooperation in the work environment.

2.2 Design as a Communication Activity

Technology has changed the traditional design process allowing the realization of complex projects in a very short time. Design teams working collaboratively are now needed to satisfy the exigency of new construction industries. Moreover complex projects require the involvement of expertise from different disciplines making collaboration and especially communication crucial factors in the process (Fischer, 1999). The designer does not work individually anymore but he/she needs to share the design tasks and specific problems with specialists from other disciplines. As Fischer (1999, p. 117) suggests "the heart of the intelligent human performance is not the individual human mind but groups of minds in interaction with each other and minds in interaction with tools and artefacts".

Due to the continue growth and expansion of human knowledge and the increasing need to specialise (Arias et al., 2000) it has become extremely difficult for an individual to solve all the aspects of a problem, "the Renaissance scholar no longer exists"(Arias et al., 2000, p. 84).

Williams and Cowdroy (2002, p. 947) have identified two main reasons for working collaboratively:

- Complexity of designing requires specialists from diverse disciplines
- A group's effectiveness in achieving design solutions is greater than that of individuals

The increasing need for communication, not only among the design team but also with clients and contractors, is evident from the growing number of experiments using project hubs. Several organisations are taking part in these projects such as W.S. Atkins, the Charter Partnership, Crest Homes and British Telecom (Brown, 2001). The purpose of project hubs is to provide information bases that are useful for all the members of the design teams in order to provide "a co-ordinated library of *managed knowledge* "(Brown, 2001, p. 129).

The use of Information Technology seems to be proving the most promising solution to filling the communication gap that may occur during the design process. Many of the organisations extensively involved in using IT as a graphical medium and for information management are now leaders in the construction industry and they are becoming centres of excellence (Brown, 2001). A price for not adapting to this change in the design process will be paid by organisations that still retain the use of traditional presentation media and information techniques (Brown, 2001).

As Shukur (2000, p. 116) pointes out "history shows that the methods of information handling are so important that they can result in changing the professional roles. [...] Architects like Brunelleschi [...] were not professional masons. [...] They had by contemporary standard superior sketching and drawing skills, which enabled them to record examples from other places, elaborate their own ideas as well as visualising not only a building but an atmosphere". The master masons were not able to produce convincing drawings for the patron. Therefore the

profession of architecture was taken over by new players who were mostly strangers to construction sites and techniques (Shukur, 2000).

As during the Renaissance, another shift is taking place and new skills are required of the designer. Delargy (1999) anticipates that IT "is shifting to managing project data, creating virtual teams and 3D intelligent modelling". The senior vice-president of the software company Bentley Systems, Etiel (Delargy, 1999) emphasises that many companies are currently abandoning 2D drawing and they are now moving towards using design software that allows collaborative working and 3D object modelling. The former head of Bovis construction's IT department, Crotty, has announced that "current stand-alone design tools fail to contribute to project management or cost management and make collaborative work more difficult" (Delargy, 1999).

Three-dimensional object-based systems will become everyday tools, and will provide "a complete interaction between design, procurement and implementation" (Brown, 2001, p. 121). 3D packages and VR tools, allowing collaboration within the design team, will be incomparably effective design tools. Crotty (Delargy, 1999) predicts that in five years time collaborative networks will be in common use and "buying a building will become as predictable as buying a car".

Further to this Atkin et al. (1996) have investigated the use of IT during the design process to enhance the communication between clients and designers. Their study shows that IT is involved during briefing and design stages but the industry at the moment is not taking full advantage of its potential. They conclude "much more emphasis needs to be placed on the early stages when Clients needs are articulated into, and interpreted into, a working design solution"(Brown, 2001, p. 39). They clarify the benefits that IT systems could bring (Atkin et al., 1996 in Brown, 2001, p. 40) as:

- Control of design information
- Better distribution of information
- Quicker turn-around of information
- Risk management, quality of information

- Cost reduction
- Increased efficiency and competitiveness

These are all extremely important points in helping the development of a project.

However, what is the role of communication within the architectural practise?

2.2.1 Architecture and Communication

The word *communication* comes from the Latin: *communicare* = to share, it essentially means "sharing by participation"(Cherry, 1985, p. 86).

Cherry (1978, p. 17) states that true communication is involved when "there is a measure of symmetry between the parties, and the messages pass to and fro. There is a continual stimulus-response, cyclic action; remarks call up other remarks, and the behaviour of the two individuals becomes concerted, co-operative, and directed toward some goal".

Concerning communication, in the field of architecture, Lawson (1990, p. 184) has pointed out that "most architects spend more time interacting with other specialist consultants and with fellow architects than working in isolation".

Broadbent (1988, p. 204) says that communication "is of vital importance to the architect because, literally, he spends almost his entire working day in tasks which in general terms may be described under the heading of communications". A study by Broadbent also involved the analysis of the various working activities of an architect sampling typical architectural behaviour (1988, pp. 205-206) (See Figure 2.1). According to his analysis an architect would spend 87.9% of his/her time communicating over various matters, making communication their most time consuming activity. Moreover, as in all design disciplines, architects are now involved in more technologically challenging projects where interdisciplinary teamwork is essential. Therefore communication and information flows are key issues in achieving quicker and more complex design solutions.

Drawing and associated activities:		
Drawing and lettering	19·2%	
Measuring	2·3	
Selecting colour scheme	1·2	
Erasing	1·7	
Searching for pencil	1·6	
Sharpening pencil	0·3	
Setting up work space	2·7	
Colouring prints	3·9	
Obtaining prints or drawing paper	0·4	
Folding prints	0·1	
		Total 33·4
Information-seeking:		
Referring to catalogues	1·2	
Referring to drawings	1·9	
Referring to specifications, bills of quantities	0·6	
Checking specialists' drawings	0·1	
Searching for drawings	2·3	
Searching for other information	0·9	
Other information away from work place	0·6	
		Total 7·6
Discussion and verbal communication:		
Discussing with colleagues or admin.	14·5	
Internal telephone	0·5	
Consulting quantity surveyor or heating and ventilation engineer	1·7	
External telephone	4·0	
Discussion with contractor, client or rep.	0·7	
On site or visiting client	8·3	
Admin. or design session	1·4	
		Total 31·1
Letters and written communications:		
Dictating letters	1·0	
Writing letters	1·0	
Signing letters	0·3	
Reading correspondence	1·0	
Making notes	1·0	
Preparing reports, specifications, financial	1·1	
Writing schedules	0·3	
Costing work, filling in time sheets or expenses sheets	0·6	
		Total 6·3
Thinking	9·5	
		Total 9·5
Miscellaneous:		
Calculating	1·0	
Walking about	2·8	
Waiting	0·4	
Personal	2·3	
Other	5·6	
		Total 12·1
		100%

Figure 2.1: Percentages of time spent in various activities in an architecture office (Broadbent, 1988, p. 205)

Despite this as Kvan (1997, p. 49) points out "the concept that sharing can be mutually beneficial is not innate" and people will need to make some effort to change their working habits.

What makes communication at the early stages difficult for an architectural design team is the natural approach of architects towards conceptual modelling. Shukur (2000, p. 136) states that "compared with for instance structural or service designers, the architect exercises very little initial analysis and jumps directly to produce a solution, a synthesis, from which he carries on". According to this approach architects often use sketches on paper to record information and to keep track of first ideas and solutions. Sketching is the architect's intimate thinking

medium, it is the predominant activity at the early phases of design and many authors have extensively studied its role and its influence on final design solutions (Lawson, 1990; Suwa et al., 1998; Do, 2001).

About sketching activities Shukur (2000, p. 35) says: "when dealing with complex information, the individual can utilise an 'intimate' medium during the production sequence/s. Such a medium assists him in downloading information from and to his brain and performing further processes like simulations, in a manner without which he may be unable to handle such information complexity. The architect's work toward his preliminary sketch paper is an example."

Although they are still widely in use in practice, traditional sketches are not effective media if concept modelling has to be shared with other participants involved in the project. Only things that can be written or drawn can be stored in a sketchpad, limiting the range of information that can be kept in mind by the designers. Abstraction and mental simulations have to be performed to successfully place the design solution in its 3D world context. Much of the material that is produced at this stage is not reusable and it is often undecipherable to any person other that the designer.

Therefore the role of traditional sketches in a collaborative context is marginal and should be substituted by more flexible and collaborative conceptual modelling tools. This exigency has already been severely felt in other design disciplines such as mechanical engineering, automotive engineering and industrial design. In fact the extreme competition that pushes these industries has made firms and companies aim to reduce the time consumed and to optimise the production development process from the very beginning of the design process (Bylund et al., 2002; Hansen et al., 2002; Cera et al., 2001). By the early 1990s Reimann and Huq (1992) had already stated that American firms, due to this increasing competitiveness, had to reconsider the way they were managing their business and they were therefore 'forced' to introduce new production methods. Among the new approaches that of concurrent engineering and Computer Supported Cooperative Work (CSCW) emerged to enhance both production and management.

These new collaborative approaches in design are mainly possible thanks to technological enhancements that occurred over the last thirty years. As Cherry (1971 p. 98) states "technology has this essential nature, that it opens up new degrees of freedom, new modes of action which, hitherto, may not have been physically, economically or even conceptually possible. Once these new possibilities are realized, then new forms of organization and operation become possible for industry, business, government and individuals".

The following sections will explain the most innovative technological advancements and the most significant phases in the evolution of communication in the Information Technology age. Communication and ease of access to information are key issues for the success of a complex project and the use of IT is proving to be the most successful approach to satisfy these crucial needs.

2.3 The evolution of Communication during the IT era

Sumerians (3000 BC), Egyptians (2100 BC) and eventually Greeks (700 BC) developed and adopted one of the greatest triumphs of the human mind: the alphabet. Leading classical scholars believe (Havelock, 1982) that this conceptual *technology* was at the base of the development of Western philosophy and of science, as it is now known. The alphabet has allowed separation of the information of the communication from the speaker making conceptualisation possible.

The spread of literacy happened over many centuries from then on, partially due to the invention of the printing press and the introduction of the paper industry. Yet it was the alphabet that was the mental infrastructure of this type of cumulative-knowledge communication (Castells, 2000). Neill Postman (1985, p. 87) summarises the quality of written communication in this way: "Typography has the strongest possible bias towards exposition: a sophisticated ability to think conceptually, deductively and sequentially; a high valuation of reason and order; an abhorrence of contradiction; a large capacity for detachment and objectivity; and a tolerance for delayed response".

Centuries later other technological achievements were involved in improving communication. Since the first introduction of the telegraph many other artificial channels have been invented which are based on systems where letters and numbers are replaced with signals. These coded communication systems needed to be learned and people had to be trained to be able to use them efficiently. Even later systems, using punch cards, were not found to have a user-friendly interface and they needed some training before use.

The telephone was the system that changed this kind of attitude. It was easy to use and accessible to ordinary people without particular training and unlike the telegraph it did not use coding and decoding cycles. In short this form of communication was very close to real face-to-face oral communication and people became accustomed to it.

Later on, the advent of television was a revolution on a massive scale. Its incredible success was clearly explained by media critic Russell Neuman, (1991, p. 103), who believes that TV became so popular "simply [because] people are attracted to the path of least resistance". Socially TV represented the abandonment of the "typographic mind", where written communication dominated. Its communication modality is seductive and its ease of use is certainly in line with Neuman's theory of it requiring very little psychological effort to decode and obtain its information. TV was a radical shift in the information broadcasting process and spread the communication of information to the masses but it has a significant limitation in that it is a one-way only communication. The audience is largely seen as one homogenous entity passively receiving information from a few centres. Contrary to this human communication is normally based on an interaction between the sender and the receiver.

A technological transformation comparable in its effect to the invention of the alphabet was achieved in this decade with the total merging of many communication modes into an interactive network. The achievement crucial to this was the formation of a meta- language, a *hypertext*, which was able to transmit, for the first time in human history, all the methods that had been developed for communicating: written, oral and audio-visual. "The potential integration of text, images, and sounds in the

same system, interacting from multiple points, in chosen time (real or delayed) along a global network, in conditions of open and affordable access, does fundamentally change the character of communication. [...] Because culture is mediated and enacted through communication, cultures themselves – that is, our historically produced systems of beliefs and codes - become fundamentally transformed, and will be more so over time, by the new technological system." (Castells, 2000, p. 357).

What made the advent of the network era possible what is now known as the "information superhighway", was mainly the introduction of computers into our everyday life. This was possible due to the newly successful combination of increased power and ease of use.

A visionary paper by Christopher Evans (1971), titled "Chatting with computers", gave an occasion to reflect on the incredible advancements in computer technology and the improvements in computer user interfaces in these last thirty years. The paper addressed a crucial topic that is still interesting nowadays but was revolutionary at the time the research was carried out: human/computer interaction. Evans analysed the use of computers in seventies society. Computers had been in use for more than 25 years at the time the author was writing, but they were far from widespread and they had a very little impact on the lives of ordinary people or even of scientists. Nowadays, technology has changed and advanced in such a way that computers have made a major impact on ordinary life in almost every field, and they have especially improved our methods of communicating. Evans (1971) also analysed the reasons why computers were not then in greater use, identifying three main issues: *cost*, *size* and *language*. Regarding cost, the author reported that the average price for a reasonably good computer during the 70s was £ 250,000. Today an entry level PC, that certainly performs better than the ones Evans was referring to, would cost less than £ 1000.

Concerning the issue of their size, the author reports that in all the public offices where computers were in use lack of space was an issue. Nowadays a computer requires a normal desk at most, and laptops and palm-tops are increasingly commonly used, which solves the space problem.

The learning curve involved in their use is not as steep as it used to be. The training period is reducing thanks to the implementation of more natural and human-friendly interfaces. Awkward and complicated machine language has been hidden behind icons and human language dialogs, making the use of computers possible even for children. The measure of this achievement can be felt when reading Evans (1971, p. 68) commenting on computer interfaces " One gets the impression that before one can make really good use of the computer - or even use it at all - one has got to have a special mathematical ability, some marvellous gift, or have been given a complicated course in programming. [...] People feel: < How can I possibly find the time to learn about computers - and even if I can find the time, am I *capable* of learning how to use them? >". Nowadays people have realised that being able to use computers is not a matter of being gifted, it simply takes a little time.

It is clear that something changed in the 1970 and it is still affecting our society. On this change Castells (2000, p. 7) says that "while technology *per se* does not determine historical evolution and social change, technology (or the lack of it) embodies the capacity of societies to transform themselves, as well as the uses to which societies, always in a conflictive process, decide to put their technological potential."

Looking at the broader picture, without the development of computer networks, large firms could never have built that complex web of strategic alliances, subcontracting agreements and decentralized decision-making processes that are now at the base of the growth of productivity. The result is the increasing need for software that helps users interact with computers in a more flexible and practical manner. From this prospective it is clear why the software industry is one of the most dynamic segments of the market, and why "the information-producing activity [...] is likely to shape processes of production and management in the future."(Castells, 2000, p. 185). Networking has now gained a crucial role in achieving organizational flexibility and business performance.

Bar and Borrus (1993) have shown that the exceptional increase in information networking technology in the 90s was mainly due to three factors: the digitisation of telecommunications, the development of broadband transmission and

a significant improvement in the performance of the computer- network system, which was mainly due to the development of suitable software and the micro-electronic industry.

The growth of network capability was possible due to the development of the telecommunication industry during the 70s and to synergic improvements in microelectronics. The revolution in telecommunication was mainly due to the introduction to the networks of electronic switches, routers and transmission technologies. ESS-1 was the first electronic switch industrially produced by Bell Labs in 1969. Less than 10 years later digital switches were introduced enhancing speed, power and flexibility, while at the same time being less energy and space consuming (Castells, 2000).

Opto-electronics (fiber optics and laser transmission) and digital packet transmission technology enhanced dramatically the capacity of the transmission lines. The new *integrated broadband networks* (IBNs) are now much more powerful than the previously revolutionary 1980s *integrated services digital networks* (ISDNs) (Lucent Technologies, 2002). These surprising improvements in telecommunication and the convergence of these electronic technologies into computer-mediated communications (CMC) together with the introduction of a transmission control protocol/interconnection protocol (TCP/IP), formed the basis for the development of the global, horizontal communication network called the Internet, which Castells considers is " the most revolutionary technological medium of the Information Age" (Castells, 2000, p. 45).

2.3.1 The Internet and its Development

The Internet originated in the 60s from a US Defense Department Advanced Research Projects Agency (DARPA) scheme to prevent the destruction or control of the American communication networks in case of a nuclear war or invasion by USSR. Paul Baran at the Rand Corporation, J.C.R. Licklider at MIT and Donald Davies and Roger Scantlebury at NPL working in parallel, conceived the underlying idea (Leiner et al., 2000). Baran (1964) officially presented his network framework at DARPA in 1964. The research aimed at the development of a network architecture

that was based on non-centralized communication technology using packet-data control which could link thousands of computers attached to autonomous networks independent from control centers. The first system developed by DARPA was called ARPANET (See Figure 2.2) and by the end of 1969 it linked only four institutions: the universities of Utah and California, in Los Angeles and Santa Barbara, and the Stanford Research Institute (Leiner et al., 2000).

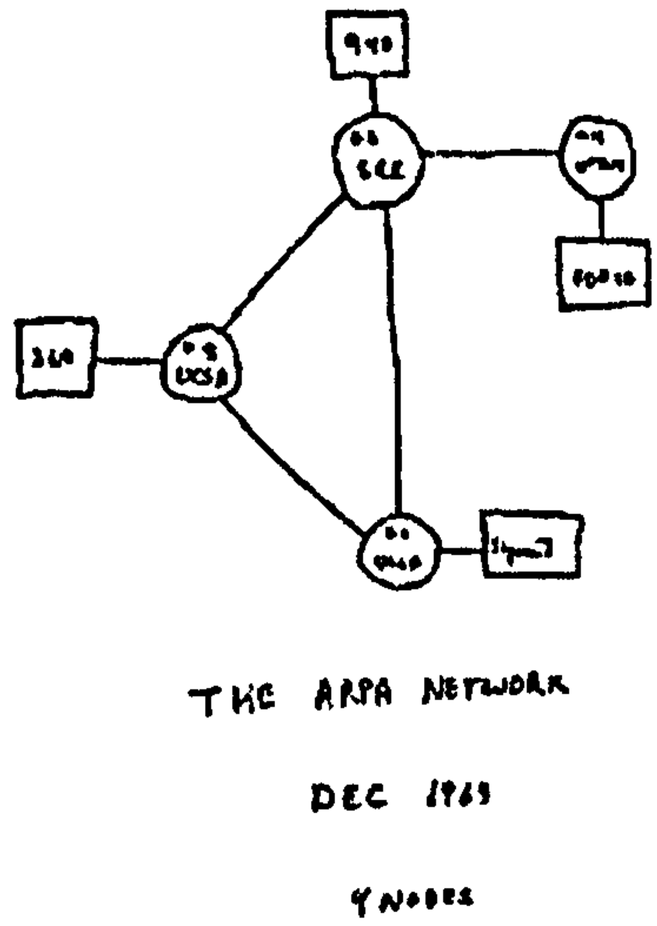


Figure 2.2: ARPANET Topology (image courtesy of Computer History Museum) (Computer History Museum, 2002)

The system was then opened up to involve research centers cooperating with DARPA and to scientists of many disciplines started to use it for all sort of purposes, from scientific communications to personal exchanges. In 1983 the original system was divided in two, ARPANET for scientific purposes and MILNET for military applications. At the same time the National Science Foundation was involved in the creation of another network for scientific research, called CSNET and together with IBM released BITNET to the wider public. The control of the whole system was still the responsibility of the Department of Defence and of the National Science Foundation, since these networks originated from the initial ARPANET system. The total system including all the networks was firstly called the ARPA-INTERNET, and then the INTERNET. After 20 years of service the ARPANET system became technologically obsolete and it was suspended on February 28, 1990. From then until April 1995, NSFNET of the National Science Foundation took over using

ARPANET as the backbone of the Internet until it was finally privatised and control by any government authority was suspended (Castells, 2000).

Many institutions were informally involved in the task of coordinating the relevant technological issues until Cerf announced the creation of what is now the Internet Society during the INET'91, the annual international meeting of CSNET participants in June 1991 (Rutkowski, 1991).

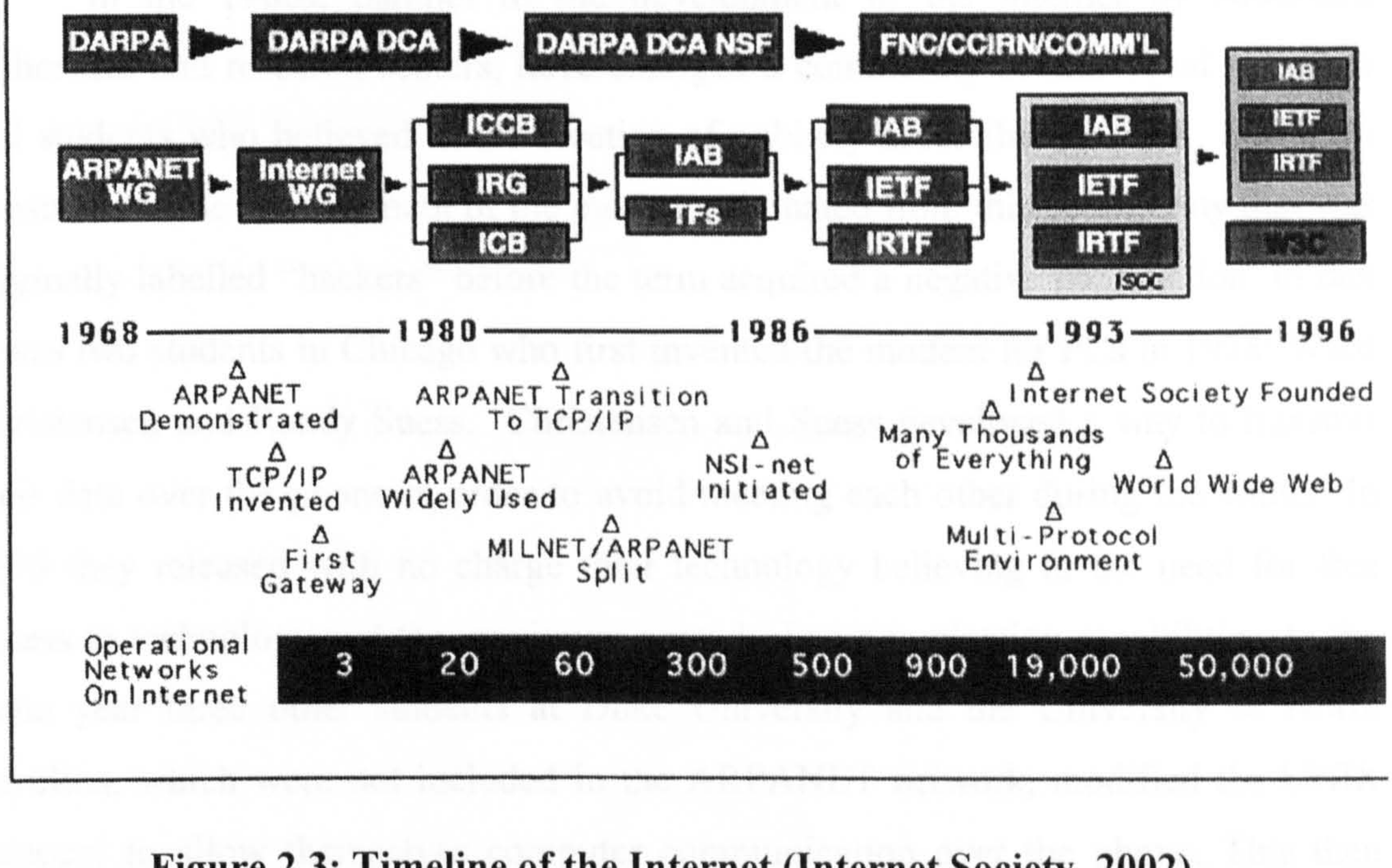


Figure 2.3: Timeline of the Internet (Internet Society, 2002)

The Internet is now a mature and reliable network system but many improvements have been required to achieve this state. In the beginning transmission at a global scale was impossible due to the lack of a common communication protocol between the different networks around the world. In 1978 Cerf, Postel and Cohen split the communication protocol into two parts: the Transmission Control Protocol (TCP) and the Internet Protocol (IP). In 1980 the resulting protocol, TCP/IP became the standard computer communication protocol for the United States. With some adaptation to European networks, the TCP/IP protocol won the battle to become the international standard protocol for computer communication (Castells, 2000).

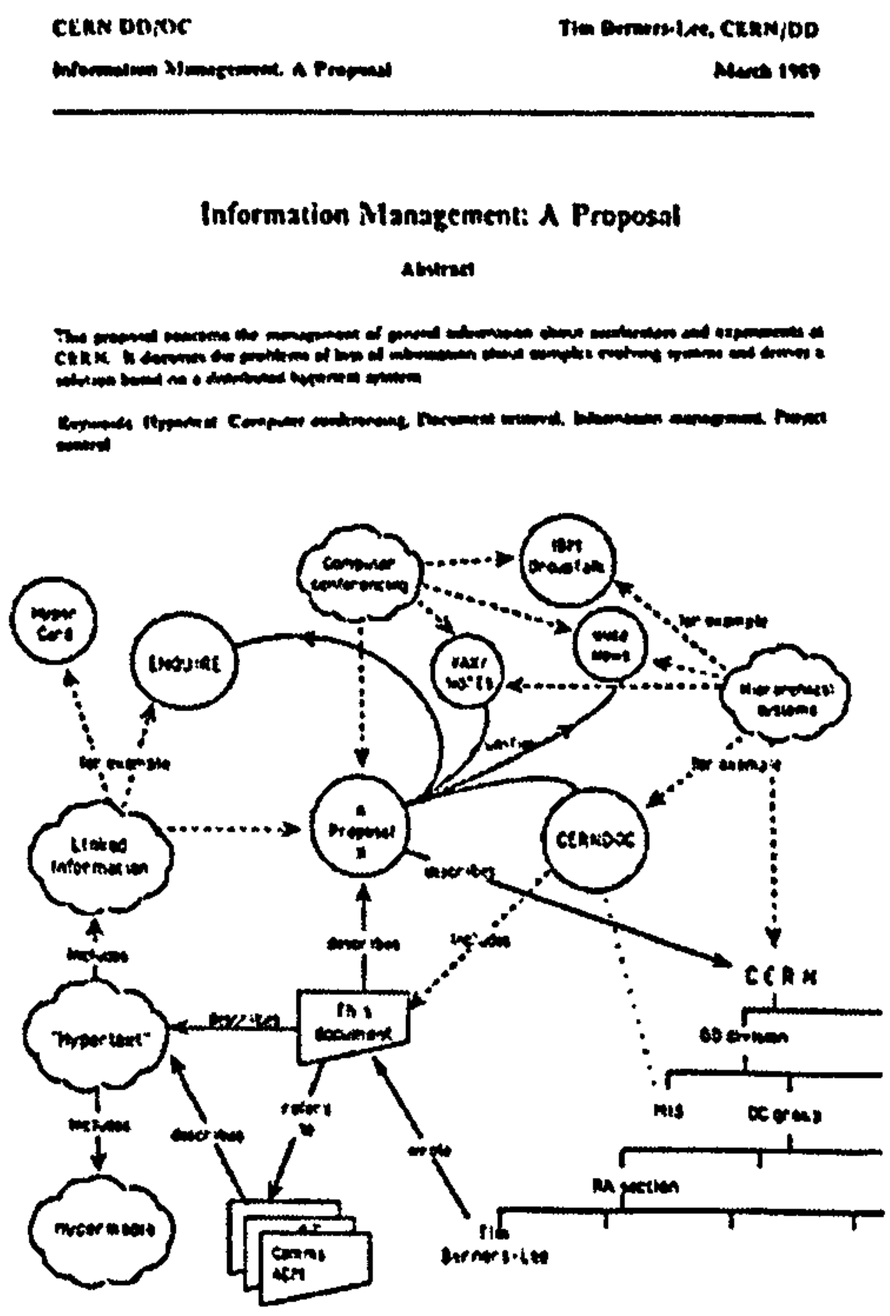
One of the first significant applications of the Internet was in allowing e-mail communication between network users. This was achieved by Ray Tomlinson at BBN, who was recently honoured by the International Academy of Digital Arts and Sciences earning the organization's ever Lifetime Achievement Award. It is now the most widespread computer communication first application in the world (Marchand, 2001).

In the 1960s, parallel to the development of the Internet by American authorities and research centers, have emerged a community of individual scientists and students who believed in the creation of publicly accessible network, free from constraints. The development of the modem originated from this community that was originally labelled "hackers" before the term acquired a negative connotation. In fact it was two students in Chicago who first invented the modem for PCs in 1978: Ward Christensen and Randy Suess. Christensen and Suess developed a way to transmit their data over the phone in order to avoid meeting each other during the winter. In 1979 they released with no charge their technology believing in the need for free access to technology and the maximum spread of communication capabilities. In the same year three other students at Duke University and the University of North Carolina, which were not included in the ARPANET network, modified the UNIX protocol to allow themselves computer communication over the phone. This then originated the on-line discussion system called Usenet which is one of the most popular conversation forums in the world today (Castells, 2000).

While in the late 1980s many companies and some millions of computer users were still communicating through alternative commercial networks without using Internet compatible protocols, by the 1990s most of them had finally adopted the TCP/IP making the Internet grow even further (Castells, 2000).

Until 1990 the Internet was still a relatively clumsy tool but the arrival of a new technology: the *World Wide Web* changed it completely. The new system (Gribble, 2001) allowed the organization of Internet sites by content rather than by location, changing the Internet into the powerful tool that we know today. The World Wide Web was invented in 1990 by the research group of Tim Berners-Lee and Robert Cailliau at the Centre Européen pour Recherce Nucleaire (CERN) in

Geneva. The same research team was responsible for the creation of a format for hypertext documents, called *HyperText Markup Language* (HTML). In their research they elaborated on the vision of Ted Nelson (Wedeles, 1966) who was active within the hacker culture of the 1970s. Nelson developed a new way to organize information within the Internet, *hypertext,* which was based on the idea of linking information in a horizontal fashion instead of vertically. A *HyperText Transfer Protocol* (HTTP) was also set up to ease communication between browsers and servers, and a standard address format, called the *Uniform Resource Locator* (URL) was implemented (Gribble, 2001).



CERN DD/OC — Tim Berners-Lee, CERN/DD

Information Management: A Proposal — March 1989

Information Management: A Proposal

Abstract

Figure 2.4: Berners-Lee's diagram describing 'hypertext', 1989 (image courtesy of Computer History Museum) (Computer History Museum, 2002a)

In 1993, Marc Andreessen, who was then an undergraduate at the University of Illinois at Urbana-Champaigne, became involved in a project for the National Center for Supercomputing Applications (NCSA), and developed the first web browser for PC's called Mosaic. Mosaic was a graphical user interface software for browsing the web. It became immediately popular due to its ease of use especially among non computer experts. In March 1994 Andreessen left NCSA and together with Jim Clark, who had created Silicon Graphics Inc. with great success, he formed

a company later known as the Netscape Communication Corporation (Gribble, 2001).

New programming languages, such as Java™ (of 1995), and Jini™ network technology (designed by Bill Joy at Sun Microsystems, Inc. in 1999) were to enrich the web even further, making the Internet a global communication network with over 300 million of users by the year 2000 - a total which is still growing fast (Castells, 2000).

2.3.2 The Internet Revolution in Communication

The Internet network evolved dramatically from its original purpose into a communication system that is suitable for any purpose by any individual or group on a global scale. On the Internet anyone can be an information provider or an information consumer. It is now accessible in almost every location thanks to the spread of the wireless technology.

Networked computer interactions are rapidly evolving from acting as mere linkages between computers to increasingly allow sophisticated cooperative systems, regardless of the geographical location of the partners involved. This has brought a new paradigm of interaction and work with computers that is gradually leading us to another social revolution. As Castells (2000, p. 394) says: "Multimedia, as the new system was hastily labelled, extends the realm of electronic communication into the whole domain of life, from home to work, from schools to hospitals, from entertainment to travel."

From a mass society, which was dominated by TV, we are evolving to a "segmented society" (Ito, 1991) where the audience is becoming more selective in choosing the information it want to access and is getting more accustomed to having an individual relationship between the sender and receiver.

Electronic mail, virtual forums, electronic journals, the World Wide Web, bulletin boards, chat lines and now collaborative working, offer great advantages if compared with traditional methods of storing and transferring information. In most cases they allow communication exchanges not provided by the traditional media and "the most important difference is [...] immediacy" (Feather, 1998, p. 108). E-mails

and chat lines are virtually instantaneous and they are more and more frequently used as new means of personal communication that is an alternative to telephone calls.

The ease of their communication is even leading us to a new written style, since e-mails and chat lines stimulate informal, spontaneous and anonymous interpersonal communication (Sulèr, 2002). Electronic discussion groups allow active participation in rather than passive reception of information. The World Wide Web assures widespread, unlimited and uncharged access to information. Collaborative work enhances interaction and sharing of tasks between users. Virtual Communities of every sort are proliferating on the Web taking full advantage of both the fast dissemination of information typical of mass media and the effectiveness and pervasiveness of personal communication. Moreover computer mediated communication can integrate many forms of expression including written and audio-visual, and it enhances the communication by allowing diversification, multimodality and versatility (Castells, 2000).

In 1994 two surveys on multimedia were published in United States. They provided information about the social acceptance of these new communication media while highlighting some very similar findings. Lou Harris (Brook et al. in Besser, 1998) found that 63% of the people he surveyed were interested in using TV or PCs to receive mostly useful information on services, business, health-care and charities etc. Within this group three-quarters of the people were interested in receiving customized news reports and asked for some form of interactive communication. 40% of those interviewed however were more interested in movies-on-demand, interactive shopping and entertainment.

Charles Piller (1994) carried out the second survey commissioned by MacWorld. He discovered that consumers desired to use the emerging networks largely for "on demand access to reference materials; educational courses or how-to programs that allow interaction with a class or instructor (distance learning); interactive reports on local schools; and access to information about government services and training". Both surveys showed the interest of people in accessing information that was not easily achievable in the past and also that personal communication facilities were desirable.

These radical technological innovations have even led to a new working paradigm that, as Dutton (2001, p. 128) states, "requires major social and organisational change".

At an individual level, the customers of commercial online systems and the Internet, which is now growing by an estimated 100,000 or more users per month, have demonstrated that they form a large market for online communication and interactive information services, as well as showing great interest in all types of educational resources. (Piller, 1994). Castells (2000, p. 393) observed that "people shape technology to fit it to their own need". Nowadays society needs communication capabilities and access to information at levels justifying the popularity of the Internet and this is the reason why it has overtaken all the other media in terms of penetration rate. In the U.S.A. it took 30 years for radio to reach 60 million users, 15 years for TV but only 3 years for the Internet after the introduction of the WWW (Castells, 2000).

2.4 The Collaborative Design Approach

Several authors (Bodker et al., 1988; Fischer, 1999; Arias et al., 2000) refer to Collaborative Design or Cooperative Design as research that analyses and investigates the possible communication and interaction between design practitioners and design end-users by means of computer-based collaborative systems. In this thesis the term Collaborative Design or Design Collaboration (Geisler et al., 2000), will be used to refer to the scientific field that investigates feasibility and provides support for the implementation of systems for remote collaboration between designers (Kvan, 1997; Laiserin, 2000a). Geisler et al. (2000) state that in design collaboration "members of project team go beyond the simple coordination of still individualistic work to engage in joint activity aimed at the co-construction of collective work products".

From universities to architectural, engineering and software firms the interest in this innovative design method is growing. This is mainly due to the advance of Internet technology and the telecommunication infrastructure that support it. Most

research on the topic has taken advantage of extensive work already done in the field of Computer-Supported Cooperative Work (CSCW) (Kvan, 1997).

In 1984 Paul Cashman and Irene Grief coined the term Computer-Supported Cooperative Work during a workshop of specialists from various disciplines who were all interested in the dynamics of group work (Grudin, 1994). Pfeifer et al. (1995), in an attempt to define CSCW, states that it is the scientific discipline that "looks at the way people interact and collaborate with each other, and attempts to develop guidelines for developing technology to assist in the communication process". Poltrock et al. (1994, p. 355) define CSCW as "a broad designation that focuses on work - the tasks that people carry out, their workspaces, and technology that does or could provide support". Grudin (1994) observed that CSCW started as the efforts of technologists to establish closer contact with experts in other disciplines such as economists, psychologists, anthropologists, organizational theorists and educators to help their research. The development of these technical systems, whose theoretical background is provided by CSCW, is addressed by *groupware* applications (Pfeifer et al., 1995). *Groupware* is the hardware and software that supports and augments group work (Pfeifer et al., 1995) or, as Poltroch et al. (1994, p. 355) defined it "software designed with groups in mind".

Desktop conferencing and video conferencing systems, collaborative authorship applications, e-mails, electronic meeting forums, collaborative applications in Computer-Aided Design (CAD), workflow management, concurrent engineering, distance learning and telemedicine are all specific areas of CSCW (Grudin, 1994), and subsequently Collaborative Virtual Environments (CVE) could be added to this list.

As discussed earlier in the chapter building construction is one of the most collaborative activities. Unfortunately so far the AEC industry has not taken advantage of the support that this technology can provide in enhancing communication (Laiserin, 2000a). Laiserin (1999b) suggests that this delay by the AEC sector is due to its intrinsic characteristics and its fragmented nature, with thousands of participants involved in the process. Laiserin (2000a) criticises that

"even the Internet, the much-heralded "mother of all collaborative environments" has been deployed in AEC to little more effect than "automating" FedEx out of the document-distribution chain". Similarly the "Web" has been used until now primarily as a passive display medium for 2D images. According to John Macomber, founder of Boston-based Collaborative Structures, Inc. (www.constructures.com) "the technology is ahead of most firms' ability to digest it" (Laiserin, 1999a).

During A/E/C SYSTEMS'99 in Los Angeles, the conference Director Brad Holtz coordinated with AEC firms and vendors to present a series of real-time project-communication demostrations (Laiserin, 1999b). According to Holtz this initiative, called the "Day in the Life" project clarified that there was a general demand for communication software from AEC practitioners (Laiserin, 1999b) in:

- Simple, proven technologies
- Affordable, off-the-shelf tools
- Web-centred project information and messaging
- Information received via each participant's preferred medium (for example email or fax)
- Real-time collaboration via whiteboarding and videoconferencing
- Team-working in parallel, rather than sequentially

Kenneth H. Stowe, Director of Project Services at Macomber, investigated the possibility of developing this new generation of software, and eventually integrated CAD tools that could introduce interoperability to the system through the Internet (Laiserin, 2000a). Stowe also created a model of the existing AEC design collaboration process in order to quantify the potential beneficial impact on cost and schedule that might occur when using alternative scenarios of improved communication (Laiserin, 2000a). On this research Stowe was working in association with researchers of the Stanford University Center for Integrated Facility Engineering (CIFE), and they developed a "project-strategy optimisation" tool called Vite Project (Laiserin, 2000a). The Vite model can evaluate more than 70 activities and the relationships between these activities within the design process (Laiserin, 2000a). The results of Stowe's pilot project indicated that about 20% of project time

and its relative cost could be saved by using Web-based collaboration. The system should allow the project participants to share the use of an integrated 3D model from the earliest stages of the design. Collaborative design can then be an efficient way to optimise and enhance the design process.

In the next section the historical evolution of CSCW and groupware will be given with examples of possible means of collaboration the effect of time and location on them.

2.4.1 Evolution of Computer Supported Cooperative Work and Groupware

In Figure 2.5 is shown Grudin's (1994) summary of the research areas and the development context for CSCW and groupware, and their evolution.

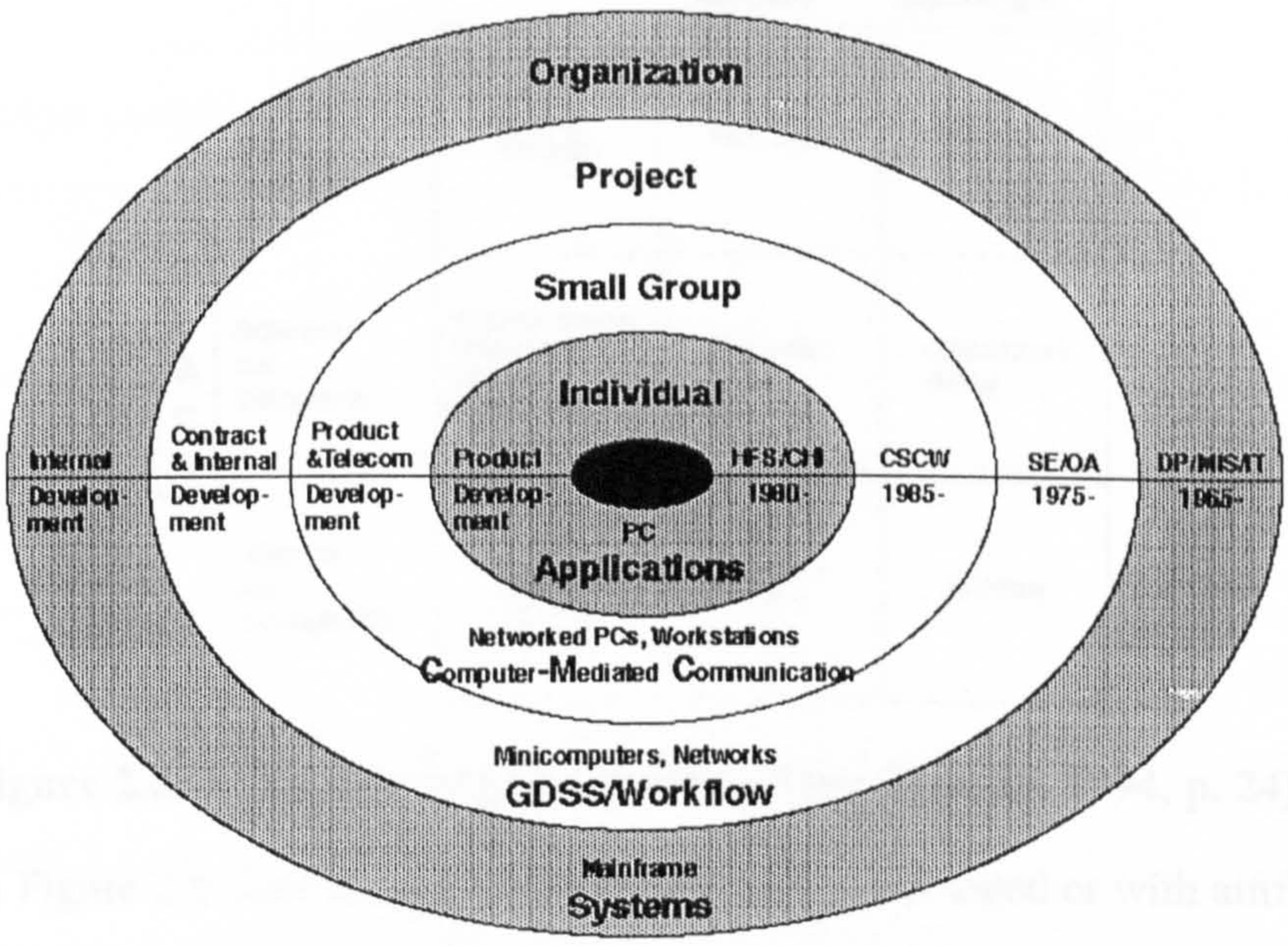


Figure 2.5: U.S. research and development contexts for CSCW and groupware (Grudin, 1994, p. 20)

The outer circle represents the introduction in the 1960s of Data Processing (DP), management information systems (MIS), information systems (IS) and information technology (IT). IS was primarily focused on organizational support and

partly on the management of large projects, and this showed the beginning of interest in studying work organization and the division of labour (Grudin, 1994). In the 1970s Software Engineering (SE) and Office Automation (OA) (now called "workflow") emerged, offering computer support for large groups and projects (Grudin, 1994). Group Decision Support Systems (GDSS) included all the technologies that contribute to decision-making, such as electronic mail and other common applications (Grudin, 1994).

The third circle represents the introduction of CSCW, during the first workshop in 1985. This was followed, in 1986, by the ACM conference on computer-supported cooperative work in Austin, Texas. The inner circle refers to individual applications and advances in Computer Human Interaction (CHI) (Grudin, 1994).

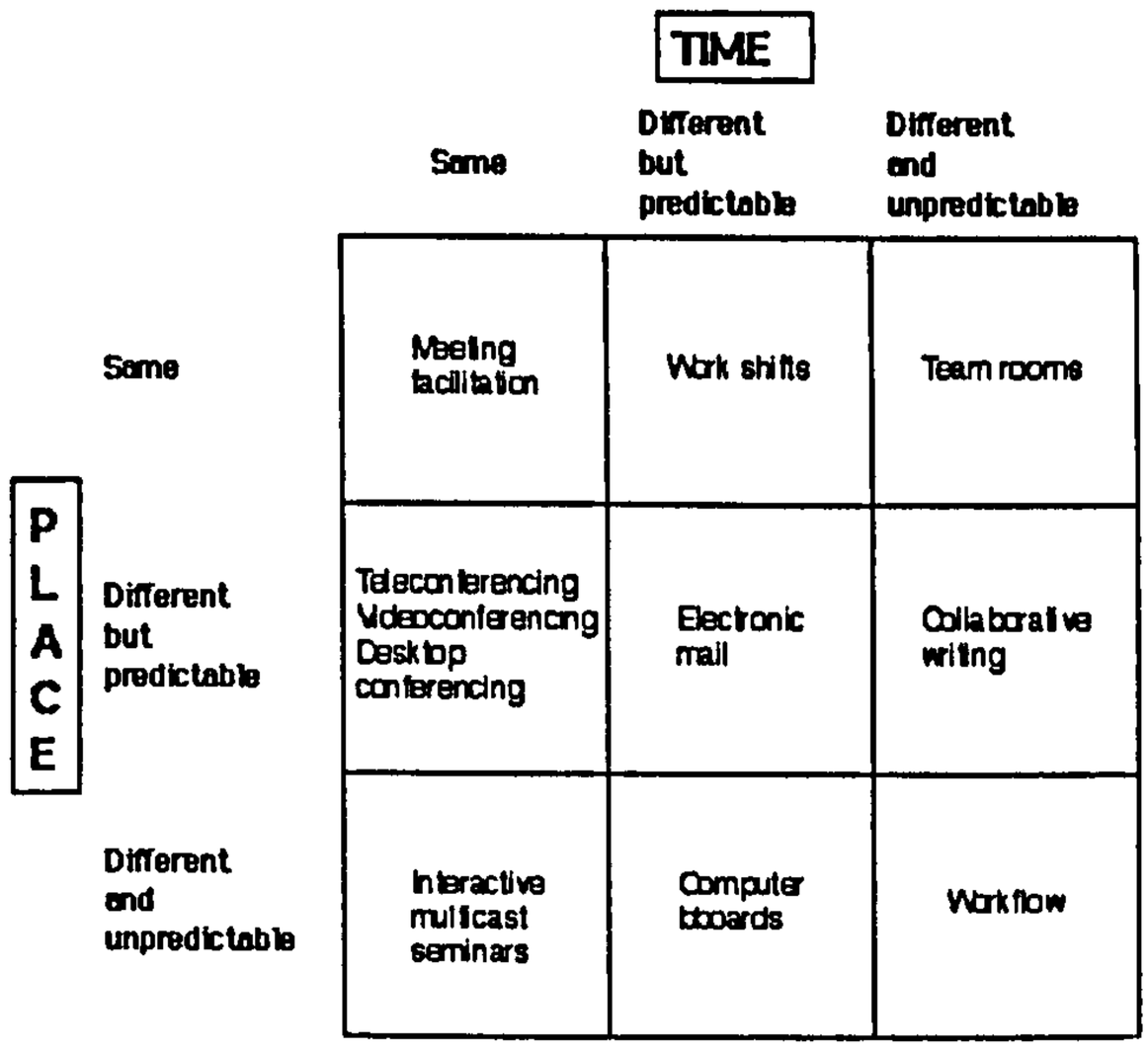


Figure 2.6: A 3x3 map of groupware options (Grudin, 1994, p. 24)

In Figure 2.6 several types of groupware are listed together with attributes of CSCW systems (Grudin, 1994). The activities are divided in a 3x3 matrix, according to the Time when the collaborative activity is carried out (X-axis) and the Place of the collaboration (Y-axis). In the diagram's left hand column are placed the activities that can be carried out in real time. In the middle column are those that can be carried out at different but predictable times, such as in the example of the limited time delay that normally occurs to reply to e-mails. In the right hand column are placed the

activities that are carried out at different and unpredictable times, such as open-ended collaborative projects (Grudin, 1994). The same type of categorisation occurs for the place where the collaborative activities are carried out (Y-axis).

Nickerson (1997) then analysed various communication possibilities and expanded the previous categories of Figure 2.6, developing a list of collaborative computing applications according to time, space and mode (See Table 2.1).

Type of collaborative application	Temporal dimension		Spatial dimension		Modal dimension			Example software
	Synch	Asynch	Proximal	Distal	Document	Audio	Visual	
Electronic messaging		X		X	X			Novell GroupWise
Information sharing		X		X	X	X	X	Lotus Notes
Document conferencing	X			X	X			DataBeam FarSite
Audio conferencing	X			X		X		Quarterdeck WebTalk
Video conferencing	X			X		X	X	Intel ProShare
Electronic conferencing	X			X	X	X	X	Silicon Graphics InPerson
Electronic meeting support	X		X	X	X			Ventana GroupSystems
Group calendaring and scheduling		X		X	X			Campbell Services OnTime
Workflow management		X		X	X			FileNet Visual WorkFlo

Table 2.1: Summary of collaborative applications (Nickerson, 1997)

Communication is divided into three parts (Nickerson, 1997):

- **Temporal Dimension**: *synchronous*, which means that activities are carried out at the same time by the participants; or *asynchronous*, meaning activities are carried out at different times.

- **Spatial Dimension**: *proximal*, which means that the group members are working in the same place; or *distal*, meaning group members working in different places.

- **Modal Dimension**: *audio*, through which a person's voice or other sounds are communicated; *visual*, through which the sight of real objects or people are communicated; *documents* or data in general, through which text, numbers, graphs, or other written information is communicated.

Nickerson's (1997) taxonomy also provides, as well as the communication types, a comprehensive list of the collaborative software that was available in 1997. Some of these packages are still on the market today, such as IBM Lotus Notes. Laiserin (2000c) refers to Lotus Notes as "the first commercially viable system to "think" outside the "personal" box of personal computing". At the beginning of the 1990s some authors based their research on it (Orlikowski, 1992). Lotus Notes has now been listed as one of the ten most important products of the decade by both Information Week and Network Computing (www.lotus.com, Wittmann, 2000).

Lotus Notes and similar applications are general-purpose communication tools. However several authors and research teams are trying to develop customised software to respond to a more specialised demand. This is the case in *Compadres* (Johnson, 2000a and 2000b) developed at the Design Machine Group, of the University of Washington in Seattle. The Compadres workgroup system was developed to address the need for a shared group workspace and it attempts to virtually reconstruct the same conditions as working in an actual architectural studio. It provides a passive presence monitor, the opportunity to share and review work-in-progress and several means of communication such as messages (Johnson, 2000a and 2000b).

Reinhard Goethert, Principal Research Associate at the School of Architecture and Planning at MIT carried out an example of the use of groupware for distal communication in urban planning in 1998. Goethert led a project funded by the World Bank that aimed to use community action planning to create master plans for several cities in Mozambique. The project was run at a distance making use of video-conferencing facilities. A series of virtual meetings were held between MIT and

Mozambique, to focus on current aspects of the planning practice in the African country. The success of the project led to proposals for other experiments including the use of distance learning techniques to give support to planners working in remote areas, video-conferencing meetings to provide suggestions and advice to practitioners and opportunity for easy access to information (Plan Newsletter, 1998).

Apart from these examples and the work of some researchers and university teams in the application of groupware in an architectural context (Christiansson, 1999; Christiansson, 2000; Modin, 1995; Jun et al., 1997; Qun et al., 2000; Hirschberg et al., 2000; Shakarchi, 2000; Tuzmen, 2000), AEC firms are still failing to take full advantage of Computer Mediated Communication (CMC) technology in their everyday work. Walter Hainsfurther, principal of Kurtz Associates in Des Plaines, and the 1999 chair of AIA's Advisory Group on Computer-Aided Practice, says: "architects want a suite of tools that is preconfigurated and easy to use. They want to manage their computing costs-hardware, software and staff-more effectively, and they want collaborative environments that help them offer more and better services. Ultimately, designers want tools that help their clients make better decisions" (Laiserin, 1999a). Laiserin (1999a) proposed an ideal set of features for the CAD package of the future. He combined all the exigencies that are shown by the AEC industry in a software package that would feature:

- Low cost and ease of use
- Easy 2D drawing output
- Navigable documents
- Intuitive solid modelling, seamless 2D/3D integration
- Broad-base standards support
- Intelligent, parametric, interoperable objects
- Lightning speed
- Integrated 3D input/output
- Photo realism
- Every available mode of flyby, drive-through and walkabout

- Universal communication capability over any type of wire or ether currently known.

AEC firms, due to the increasing pressure for attaining quality and the shortening of the time allowed for project development, will have to investigate several means of communication to enhance the relevant design processes. The sharing of information in these firms does not only involve the public domain of the Internet but also extends to IP networks within and between enterprises (Laiserin, 1999b). The term Intranet, coined in 1994 by the software analyst Steve Telleen (Telleen, 1995), means an IP network accessible only by authorized people, for example the employees of one firm. Booz Allen and Hamilton Inc., a major management-consulting firm, has used Intranet technology since 1995 to provide its 6000 employees with access to information and documents supplied by 80 offices around the world (Dutton, 2001).

The theory of the extranet was developed in 1996 by the inventor of Ethernet Bob Metcalfe (Kirsner, 1998). It is an intranet that has been opened to selected external businesses, firms and clients, through a secure IP connection (Laiserin, 1999b). Because these sites can be hosted by Web-servers within the firms, in AEC contexts they are now referred to as project-specific Web sites, project Webs or more commonly as project collaboration networks (PCNs) (Laiserin, 1999c). Within a PCN clients can browse documents and view CAD files on-line without being obliged to install a suitable CAD package on their computers (Laiserin, 1999c). Most PCNs include memos, notices and message features and they also have pan and zoom tools to help to navigate documents. Leading firms such as Bechtel (www.bechtel.com), Zimmer Gunsul Frasca (www.zgf.com), Parsons Brinckerhoff (www.pbworld.com) began developing communication tools in order to explore the impact of more efficient communication on the design process (Laiserin, 1999b). Lawrence Rocha, chief information officer of Wimberly Allison Tong & Goo, predicts that PCNs "are paving the way for centralized building-data models. This technology provides a foundation, a single place, where intelligent building data can exist and evolve along with the project. Over the next couple of years every sizable project will require data-centric project sites that allow design teams to collaborate more effectively" (Laiserin, 1999c).

Nevertheless these PCN services require a certain degree of conscious effort to upload and publish all the documents that are in progress to the Web, making the consistency of information on these systems an important issue. To solve this problem the general approach is to integrate and link the software used for design generation, such as CAD packages, with software that allows the sharing of drawings and documentation among all the participants in the process (Laiserin, 1999c). According to Laiserin (2000b) in manufacturing design the use of collaborative environments, evolved from model-based tools, CAD packages or developed by CAD tools makers, is already well established. This approach has a 3D model as the focus of the collaboration, and the communication tools are integrated in software that is already a familiar interface for the users. 3D models can be shared over the Web allowing the participants to interact in real-time and concurrently with other designers and with the model. Examples of these packages are IronCAD (www.ironcad.com), Alventive (www.alventive.com), Alibre (www.alibre.com) and Cooperative ARCADE, developed at the Fraunhofer Center for Research in Computer Graphics, Inc. In AEC the software for collaboration did not evolve from existing CAD packages but was mostly originated by management consultants (Laiserin, 2000b). The result is that AEC designers still communicate via a discontinuous and asynchronous exchange of 2D drawings which makes the collaboration time consuming and less effective than it could be. For Laiserin (2000b) the real success of design collaboration in AEC is dependent on the decision of CAD vendors to make their 3D modelling tools compatible with Web-collaboration. "When designers can collaborate with the rest of the project team while staying within their familiar design interface [...] then AEC design collaboration will follow the success of mechanical and manufacturing design collaboration"(Laiserin, 2000b).

According to Laiserin (1999a) the next step in collaborative design will involve yet more challenging technology. Don Weinreich, senior associate at Polshek Partnership Architects LLP of New York, complains that the tools available at the moment "deliver only 20 percent of what we'd like to have" and he argues that "we need to get rid of the keyboard and the mouse, and we need much larger displays

with much finer resolution if computers are to become as natural for architects as pencil and paper" (Laiserin, 1999a).

One possible approach to address these emerging needs in the design and AEC industries could be the use of virtual reality tools and devices which are now also able to secure collaboration and concurrent presence over the Internet. Collaborative Virtual Environments (CVEs) are, at the moment, the ultimate tools to simulate, visualise and interact in 3D shared environments.

2.5 Conclusions

This chapter highlighted various aspects of communication and collaboration activities and their role in modern working practices. The Internet and the increasing growth of network capabilities have brought a great change to the work environment and its management. Advances in technology have led to a new working paradigm where teams are geographically dispersed and activities and information are shared over the net. The AEC industry is starting to experiment with these new challenging possibilities in order to enhance quality and shorten the time needed for the realization of the final design product. Several firms and institutions have developed collaborative systems for concurrent design, collaboration and exchange of information. Unfortunately it has been difficult to establish a common working practice resulting in a very fragmented use of these new collaborative applications. Architects complain of the inadequacy of most of the software available at the moment, as being mainly focused on document management rather than on 3D modelling, but they look with growing interest at the future possibility of employing the new technologies and visualization devices to fill this gap.

The next chapter will be highlighting the use of Collaborative Virtual Environments (CVEs) as a new means of collaboration and some examples of CVEs systems will be presented.

3 The use of VR in Collaborative Design

3.1 Introduction

The last chapter highlighted the importance of communication, information sharing and collaboration in design activity and especially in the architectural practise. The latest achievements in CSCW systems, concurrent engineering and distributed working systems have enhanced productivity, the exchange of information and work dynamics. As the scale of the project increases the sharing of the workload becomes even more crucial especially when remote parties are involved.

This chapter moves towards the development of collaborative systems and related technology and it will focus on the use of Virtual Reality (VR) as the new means of collaboration. The features of Collaborative Virtual Environments (CVEs) are outlined and there will be an overview of CVEs systems, from the earliest to the most recent Collaborative Virtual Design Environment (CVDE).

3.2 Virtual Reality Technology

Nowadays computers are opening up possibilities that were unimaginable in the past and this revolution is changing the way we relate, communicate and possibly work, with others.

Authors in the 1980s felt the excitement that exploring these new frontiers offered. One of these was William Gibson and with his book *Neuromancer* (1989) he is considered by many to be the father of *Cyberpunk*, which is not only a body of literature but also a popular cultural movement.

The *cyber* aspect of cyberpunk stories is usually the existence of a *system* that dominates the lives of the ordinary people involved. These systems are promoted by several technologies and particularly by information technology. Often the technological system extends to involving human *components* and humans themselves become part of "the Machine" (Schneider, 1996).

From the revolutionary Cyberpunk movement, William Gibson coined the term *cyberspace* meaning an "*information-space* loaded with visual cues and navigable with brain-computer interfaces called *cyberspace decks*" (source The Free On-line Dictionary of Computing www.Dictionary.com, key entry: *cyberspace*).

The author of the Autodesk Cyberspace Project, Randy Walser, proposed another definition: "Cyberspace is the medium that gives people the feeling they have been transported, bodily, from the ordinary physical world to worlds purely of imagination" (Walser, 1992 in Kramarae, 1995, p. 38).

Some authors use the terms *Cyberspace* and *Virtual Reality* as synonymous although *Cyberspace* usually refers to the global network of computer-mediated-communications (CMCs) where graphic worlds and inter-relationships coexist in the same imaginary space (Kramarae, 1995).

Virtual Reality allows the interaction with virtual worlds that Dagit (1993, p. 513) defines as "multidimensional, interactive, computer-generated environments, which enable people to act in and upon a space with varying degrees of real-time interaction, as they would in the real world. These systems allow the user to enter a highly effective information environment, which heightens data awareness and understanding, and thus leads to enhanced productivity."

France was one of the first countries where interest was shown in Virtual Reality (Burdea et al., 1994). The first major international conference on the topic, drew hundreds of papers, was organized in Montpellier in March 1992, and had the title *Interfaces for real and virtual worlds* (Burdea et al., 1994).

Another major event followed in San Diego, where the first conference on *Medicine Meets Virtual Reality* was organized (Burdea et al., 1994). The conference was the occasion for 180 doctors and 60 scientists to meet and discuss the potential applications of Virtual Reality in medicine. After these conferences the first scientific journals were published and the first books on the topic appeared (Aukstankalnis et al., 1992; Pimentel et al., 1993).

The turning point for the development of Virtual Reality technology was the involvement of the world's largest professional society, The Institute of Electrical and Electronics Engineers (IEEE), which organized its first conference on VR in

Seattle in September 1993 (Burdea et al., 1994). VR was then included in the mainstream of scientific research and it is now one of the most challenging topics of the engineering community (Burdea et al., 1994).

In 1992, which was the same year of the first international conference on VR technology, the Electronic Visualization Laboratory (EVL) of the University of Illinois, Chicago, presented an immersive projection system called CAVE (CAVE Automatic Virtual Environment) (Cruz-Neira et al., 1993) at SIGGRAPH'92, the Annual International Conference on Computer Graphics and Interactive Techniques.

The introduction of this revolutionary system was a turning point for the technology due to the amazing improvement of the sense of presence experienced by the user. Thanks to the highly immersive displays, the high resolution of the images and the least physically awkward hardware and equipment available at that moment, no system could compete with the CAVE in effectiveness and feasibility for commercial applications (Dagit, 1993).

In fact, before the introduction of the CAVE, most of the VR systems available were based on Head Mounted Displays (HMDs) for stereovision and they used gloves to interact with the environment (Dagit, 1993). Despite this experiment in exploring the new frontiers that VR was opening up it was hard to use the equipment due to the high degree of physical discomfort due to the HMD and this affected practically the exploitation and commercial use of the technology. Furthermore most of the systems available had very poor resolution and did not allow multi-user capabilities (Dagit, 1993). The layout of the CAVE however permitted several users on the system concurrently and this innovation was of dramatic importance for the commercial application of the technology (Dagit, 1993).

Today VR is a broad field with a multitude of companies and research bodies actively involved in developing, producing and studying software, hardware and theory for VR technology.

VR is a way to experience worlds that are not necessarily real or material. Therefore it gives the user the freedom to simulate and eventually build up knowledge and skills that would otherwise be too dangerous, too expensive or simply

impossible for human beings to achieve. VR enhances users' simulation capability by providing (Dagit, 1993):

- **Immersion:** Users perceive themselves to be completely surrounded by the information environment.
- **Presence:** The participant actually has the sensation of being *in* the environment. The Virtual Environment (VE) is a place on its own, with different characteristics compared to an ordinary environment. Real world constraints can either be easily simulated, i.e. gravity or collision detection, or removed to provide impossible experiences to the users, i.e. flying or viewing microscopic particles as if they were of a size comparable with human beings.
- **Interactivity:** This is surely the most important feature that VR gives its users. The environment involves the participant through the real time feedback which follows each user's actions.
- **Autonomy:** Participants are neither constrained in paths or views preset by others but they have the freedom and autonomy to explore every part of the environment.
- **Collaboration:** Multiple users are able to take part and they can interact in the very same VE. These systems are now referred to as Collaborative Virtual Environments (CVEs). Different levels of collaboration are available in CVEs, depending on the systems adopted, but the most common manner of collaboration is users co-existing in the same VE.

The collaborative dimension of virtual worlds is a fast growing research area and it is the main topic of this thesis. Morningstar et al. (1990) consider this aspect so important that they state that "at the core of *our* vision is the idea that cyberspace is necessarily a *multiple-participant environment*".

The next sections will describe in detail what a Collaborative Virtual Environment is and will present several types of CVEs with a variety of purposes.

3.3 Collaborative Virtual Environments

Snowdon et al., (2001) observe that books such as *Neuromancer* (Gibson, 1989) describe the Collaborative Virtual Environment (CVE) as "distributed virtual reality systems that offer graphically realized, potentially infinite, digital landscapes. Within these landscapes, individuals can share information through interaction with each other and through individual and collaborative interaction with data representation" (Snowdon et al., 2001, p. 4).

The authors (Snowdon et al., 2001, p. 4) consider this definition too restrictive and they suggest that "a CVE is a computer-based, distributed, virtual space or set of places. In such places, people can meet and interact with others, with agents or with virtual objects. CVEs might vary in their representational richness from 3D graphical spaces, 2.5D and 2D environments, to text-based environments. Access to CVEs is by no means limited to desktop devices, but might well include mobile or wearable devices, public kiosks, etc."

CVEs provide a space that contains data representation and users. Individuals and data inhabit the VEs, the first in the form of embodiments or avatars, the second mostly as geometrical 3D representations. Furthermore CVEs combine the graphical representation potential of VR technology with communication capabilities proper as seen in more traditional CSCW systems. Therefore VR brings several advantages to CSCW, such as:

- The ability to present a large amount of information
- Three-dimensional representation of data
- Support for many sensory modalities
- Natural multi-user interaction
- Awareness of co-workers' activities.

Thanks to their flexible capabilities CVEs are increasingly being used for a variety of tasks, from military and industrial team training, collaborative design and engineering, education and social behaviour study, to entertainment as in multiplayer games.

According to Snowdon et al. (2001) software for CVE systems should support some key features, such as shared context, awareness of others, negotiation and communication, flexible and multiple viewpoints, and the following paragraphs will describe these concepts in detail.

3.3.1 Shared Context

Shared context is a crucial feature in CVE systems and Snowdon et al. (2001) stress that its meaning can vary and several interpretations can be provided according to the specific application. The authors (2001, p. 9) define shared context as "shared knowledge of each other's current activities, shared knowledge of other's past activities, shared artefacts and shared environment. Together, these lead to shared understanding".

In practice, the most common approach in CVEs is to provide the users with a shared space where they can accomplish shared activities through gestures that reveal their intentions. Shared artefacts should be visible and available to negotiate and the current focus of the users attention should be indicated in order to help communication. As Snowdon et al. (2001, p. 9) suggest "when artefacts are shared, not only do they become the subject of communication between users, but also the medium of communication; as one user manipulates an object, changes to it are visible to others in an externalisation of the processes of change".

3.3.2 Awareness of Others

Dourish et al. (1992, p. 107) define awareness as "understanding of the activities of others, which provides a context for your own activity".

In this regard CVEs adopt a number of metaphors to achieve group awareness. Most commonly users are represented in the virtual scenario through virtual embodiment, in the literature this is described as *avatar*, which inhabit the VE and represent the position of the users in the virtual scene.

Capin et al., (1998, p. 158) list the most important functions of avatar representation in CVE:

- Perception of other participants
- Awareness of users' location in the VE
- Identification
- Manifestation of participants' interests
- Visualization of participants' actions.

In short avatars are acting as links between participants and the VE. Their use enhances a person's psychological involvement in the virtual environment thus intensifying the relationship among users within the environment.

In CVE it is crucial to differentiate the virtual representations of the participants from the rest of the VE at a glance. Human shaped avatars are clearly different from the objects in the VR scene thus the human metaphor is commonly used. Nevertheless avatars can be of various types, according to the metaphor that the CVE application embraces and the degree of realism that they are supposed to achieve: from human shaped 3D models to 3D hands or faces, animals or objects, or even video avatars.

3.3.3 Negotiation and Communication

Communication among the participants in CVEs can be of a verbal or non-verbal nature, and it is established through a multitude of channels, from textual to auditory and/or visually.

Textual communication is often by means of virtual chat boards or white boards. Auditory channels involve the use of voice-conferencing facilities and equipment, such as microphones and loudspeakers. Visual channels embrace both the movement and behaviour of the avatars in the VE and/or video streaming for video conferencing facilities.

Some applications, when focused on the achievement of realistic virtual environments, aim at recreating face-to-face (f2f) natural communication among the participants by using avatars and video conferencing facilities. This approach involves many aspects of human communication such as facial expressions, lip

movements, body postures and gestures. These redundant communication channels contribute to the achievement of better comprehension and clarity.

According to Capin et al. (1998, p. 158), these more natural representations of the participants in virtual contexts have the following qualities:

- The visual embodiment of the user by means of avatars
- The means of interaction with the world is through the free movement of the avatars
- The means of feeling the various attributes of the world by using the senses. This is achieved through mutual awareness among participants and a sense of proximity through avatar representations.
- Enriching the communication to increase understanding among participants when using video conferencing facilities.

Even though f2f implies enriched communication, it is difficult to achieve due to the large amount of data required. A variety of data from different sources has to be streamed across the network such as captured audio and video and data representing tracked gestures and body postures.

For computer-supported collaborative work applications and especially CVEs, the closer proximity to face-to-face communication felt among participants the better the applications allow group work and collaboration.

On the other hand traditional computer chat features, hiding the real identity of the users, can have the effect of increased interaction as this can eliminate shyness and embarrassment.

Suler, in *The Psychology of Cyberspace* (2000), addresses the psychological issues related to cyberspace communication. He noted that in chat-based cyberspace communications people are free to express their identities fully, partially, or not at all and they have the means to hide or disguise themselves and they can eventually achieve what the literature describes as *Identity Flexibility* (Suler, 2002).

3.3.4 Flexible and Multiple Viewpoints

In the traditional CSCW paradigm viewpoints are often managed following the simple WYSIWIS (What You See Is What I See) concept. According to this metaphor, users are forced to the same view of the document they have to share (Valin et al., 2001).

CVEs have demonstrated a need for more flexible multiple viewpoints allowing all the users independent views and providing participants with flexible viewpoints according to the task that they have to accomplish (Stefik et al., 1987 and Valin et al., 2001).

The WYSIWIS approach was found to have limits dealing with two-dimensional collaborative applications, such as shared whiteboards, but its use became really inadequate in CVEs where multiple participants interact independently in the virtual scene (Stefik et al., 1987 and Valin et al., 2001).

3.4 Collaborative VR Systems

The previous sections introduced the concept of CVE and summarised the most important issues related to its implementation. Following this, the next paragraph outlines a classification of CVE systems according to five main categories, which depend on the systems' architecture and features:

1. Large distributed virtual simulators
2. Text based shared VE: MUDs
3. Multiple user navigation systems
4. Platforms for implementing VR shared applications
5. Collaborative Virtual Design Environments.

This categorization excludes the examples of tele-presence and augmented reality that are out of the scope of this thesis. In the following paragraphs each of the five categories will be explained in detail and examples will be provided.

3.4.1 Large Distributed Virtual Simulators

The military research into Distributed Simulations applications, carried out in the United States by the Defence Advanced Research Projects Agency (DARPA), was a central contribution to the development of the history of shared virtual environments.

In 1983 the United States Army (McCarty et al., 1994) started a project called Simulator Networking (SIMNET) with the goal of developing a wide-area network, involving hundreds of simulators exchanging information over a communication network.

In 1989 DARPA and the Army Project Manager for Training Devices (McCarty et al., 1994) started a project to develop a more efficient protocol for simulation known as the Distributed Interactive Simulation (DIS) standard. The DIS was based on SIMNET but its technology went beyond this early distributed application thanks to advances in communications technology (McCarty et al., 1994).

The DIS was essentially a group of standards addressing the relevant topics, such as (Macedonia et al., 1994):

- Communications architecture
- Format and content of data
- Entity information and interaction
- Simulation management
- Performance measurement
- Radio communications and emissions
- Field instrumentation
- Security
- Database formats
- Fidelity
- Exercise control and feedback.

It also provided specifications to be used by the government agencies and engineers building simulation systems.

In Figure 3.1 is an overview of the first distributed applications. It is to be noted that research towards military simulators and research by the computer games community were at the base of the early technological achievements in distributed applications (Fujimoto, 2002) (See Section 3.4.2).

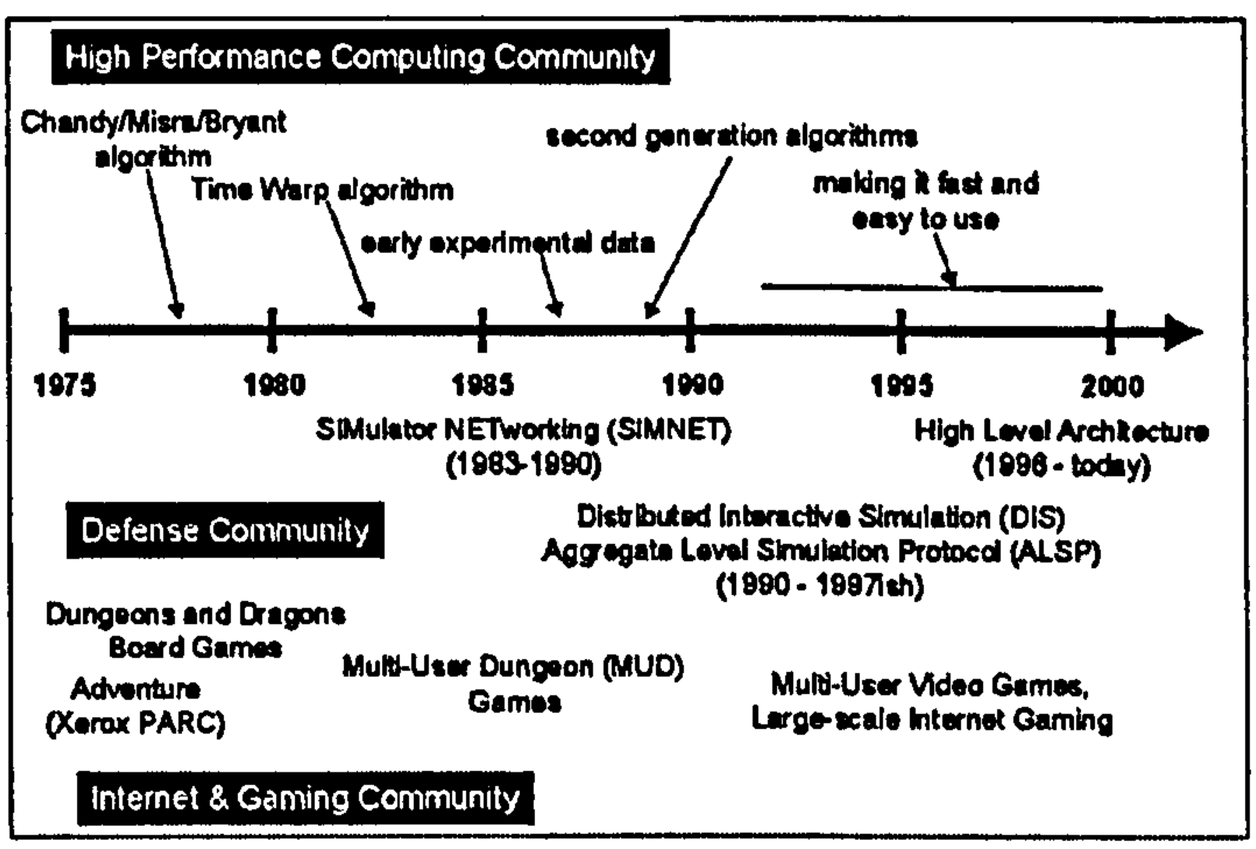


Figure 3.1: Historical overview of Distributed Computing and Simulation (Fujimoto, 2002)

The following two sections will outline two examples of virtual simulators: The Virtual Cockpit and the NPSNET system.

3.4.1.1 The Virtual Cockpit

In 1991 DARPA commissioned the Air Force Institute of Technology (AFIT) to develop a low-cost flight simulator to be used for distributed simulations (McCarty et al., 1994). The outcome of the research was published by McCarty et al. (1994) in a special edition of IEEE Computer Graphics and Applications on Virtual Reality.

AFIT developed a virtual cockpit consisting of a Silicon Graphics workstation connected to a SIMNET simulator system which had a head –mounted

display device and position trackers. The virtual cockpit had the fundamental capability to broadcast and receive network distributed simulation messages. Messages were transmitted which had the information required to upgrade the current position of all the participants in the simulated scenario in real time. The system had to control, in real-time:

- The flight dynamics and cockpit instruments
- The display of the virtual scenario
- The network interface.

The software for the flight dynamics was based on a mathematical model developed at the Naval Postgraduate School (NPS) by Major Joe Cooke (Cooke et al, 1992). This numerical model incorporated data such as the virtual aircraft's speed, position, velocity and acceleration, in the simulated environment.

The display of the virtual scenario required two major functions: database construction and database rendering. The database was built using Software System's MultiGen, a commercial polygon-based modelling package that stores descriptions of geometrical objects in its proprietary format, called Flight. The database of the VE consisted of terrain, stationary entities and dynamic objects.

The rendering of the database was achieved through minor changes to the code using the Silicon Graphics graphics library, GL.

Networked communications to and from the virtual cockpit involved the transmission of the current state of the vehicle. This included information such as the location, orientation, speed and appearance of the vehicle e.g. if it was on fire or damaged etc.

In terms of software implementation the network interface had two main parts: a set of Unix-style network daemons that provided access to the Ethernet messages contents and a C++ class, called the "entity object manager" that handled the content of the messages and maintained the description of the simulated environment.

The entity object manager also performed remote vehicle approximation of simulated objects, known as *dead reckoning*, used to reduce network load. Dead

reckoning allows interpolation of the intermediate positions of the remote participants, based on the last received network message. During their virtual motion remote vehicle simulators do not transmit every change in their position but they broadcast their position and orientation only when these cannot be predicted. This reduces considerably the amount of data to be sent through the network.

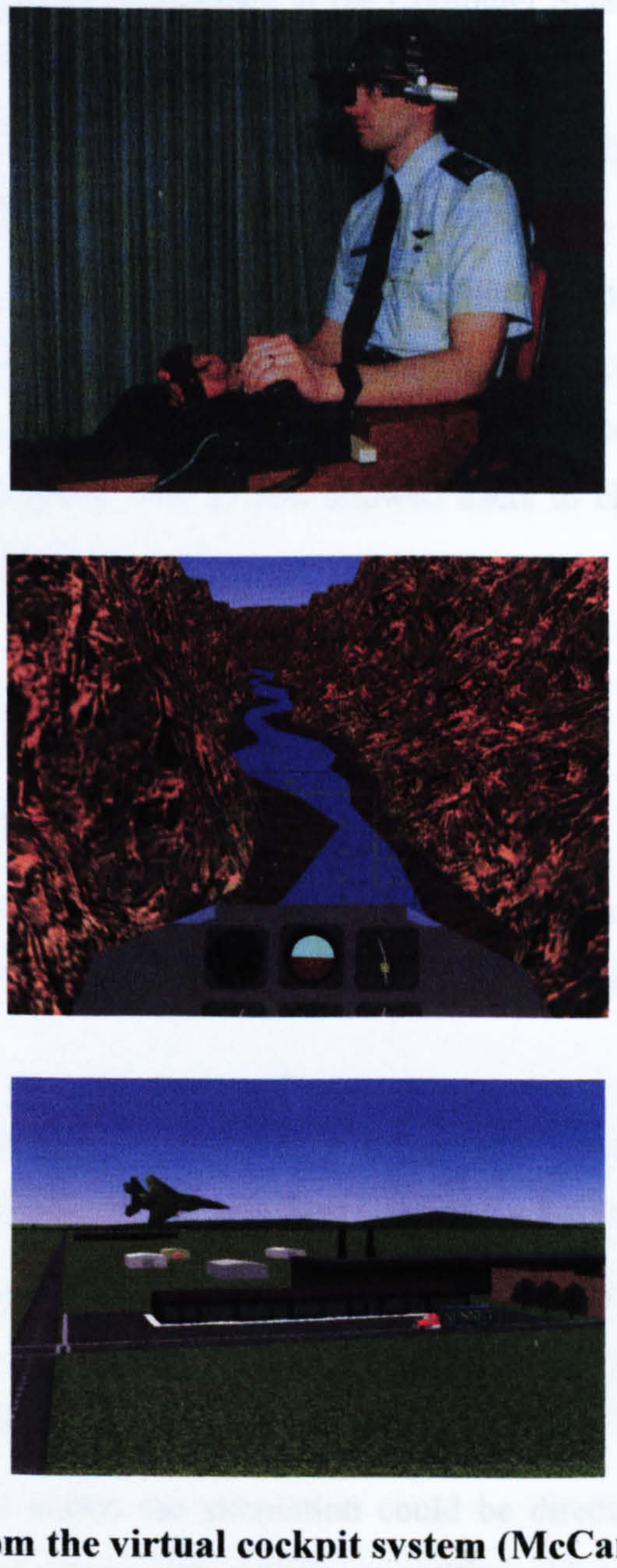


Figure 3.2: Images from the virtual cockpit system (McCarty et al., 1994, pp. 50, 55)

3.4.1.2 NPSNET

NPSNET (Naval Postgraduate School Networked Vehicle Simulator) (Shaffer, 1995) is an ongoing project that investigates the implementation of a networked, distributed simulation system which allows multiple users to interact within the same virtual environment. NPSNET is a low cost, real-time networked vehicle simulator which was developed at the Computer Science Department of the Naval Postgraduate School (NPS) in Monterey, California.

The current, still under developement, version of this framework is NPSNET-V (NPSNET-V, 2002) that follows the succesfully completed, NPSNET-IV project.

NPSNET-IV (Macedonia et al., 1994) was the first 3D virtual environment that incorporated both the IEEE 1278 Distributed Interactive Simulation (DIS) application protocol and the IP Multicast network protocol for multi-player simulation over the Internet. The system allowed users to choose between several simulated environments:

- Air
- Ground
- Nautical, surface or submersible
- A virtual vehicle, which is a non-invasive entity that exists in the VE, but it is not represented in the virtual scene by a virtual model e.g. a stealth vehicle
- Human

Users could manoeuver the vehicle using three interface devices: a flight control system, a six degree of freedom SpaceBall and a keyboard.

In this simulation configuration the virtual vehicle, provided with munitions to be used during the combat simulation, could move on the ground surface, on the sea surface, below the sea surface or in the atmoshere.

Other vehicles within the simulation could be directly controlled by other participants, rule-based autonomous entities, or entities with scripted behavior.

The virtual environment created by NPSNET-IV included a variety of static and dynamic objects exhibiting numerous visual qualities and multimedia behaviors including sound, flying over a farm for instance evoked a chorus of animal noises.

NPSNET-IV was mostly written using Performer, a mid-level Application Programming Interface (API), created by SGI, which performs the movement and viewpoint computations automatically (Macedonia et al., 1995).

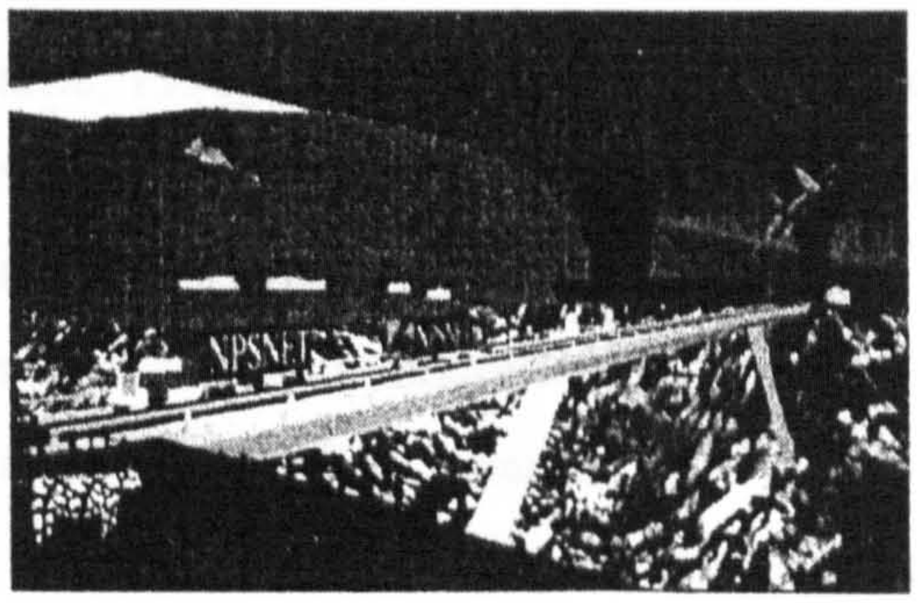


Figure 3.3: An image of the NPSNET simulator scenario (Darken et al., 2001)

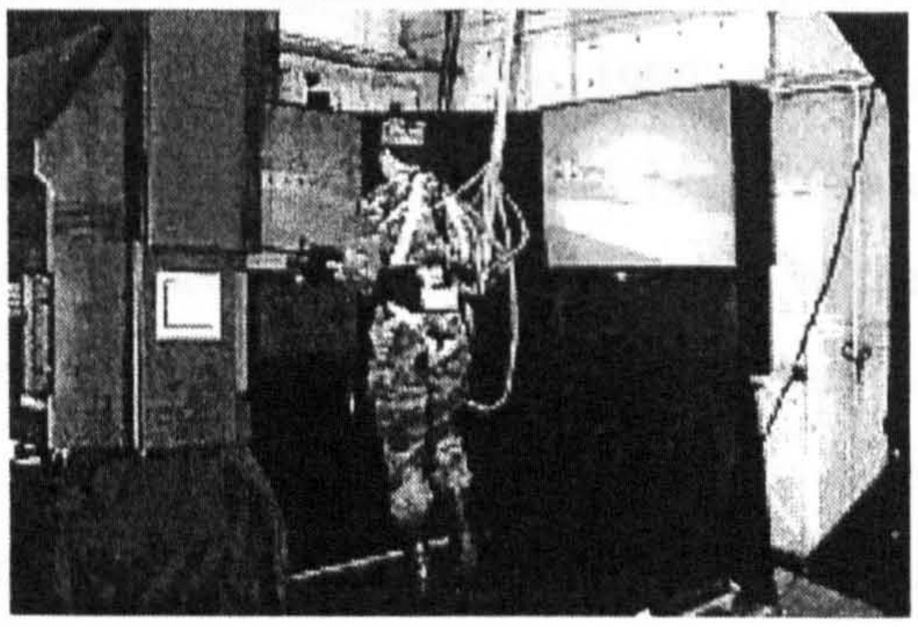


Figure 3.4: A soldier being trained and his virtual embodiment using NPSNET (Darken et al., 2001)

3.4.2 Text-based Shared VE: MUDs

MUDs come from a very special family of multi-user computer programs. MUD originally stood for Multi-User Dimension, the name given by Richard Bartle and Roy Trubshaw (Reid, 1995). Now the term most commonly refers to Multiple User Dungeon or Multiple User Dialogue: it is a networked, multi-participant system mostly found on the Internet.

Roy Trubshaw (Bartle, 1990) wrote the very first MUD programme in spring 1979, while he was student at Essex University. The game was originally little more than a series of inter-connected locations where the users could move and chat to each other. Trubshaw rewrote it almost immediately, and the next version was much more sophisticated, but also called MUD (Bartle, 1990). The database (rooms, objects, commands etc.) was defined in a separate file, but it could be added to during the game. Due to concerns over its memory and because this first version of the programme was becoming unmanageable Trubshaw rewrote it from scratch. The new release was completed by Easter 1980 (Bartle, 1990). This version, (which is in fact version 3), is now considered as the original MUD (Bartle, 1990). At the beginning of the 1980s Essex University, USA, was linked to the ArpaNet through an experimental packet-switching system, called EPSS (Bartle, 1990). This link offered the chance to test the program with external players. In the December 1980 issue of Byte the first reference to MUD appeared in print (Bartle, 1990). MUD software was then developed and maintained by Richard Bartle (Bartle, 1990).

Jim Aspnes (Burka, 1993), a graduate student at Carnegie Mellon University, USA, wrote a game called TinyMUD over one weekend of August 1989. It was a simple, multi-user game that was available to anyone via the Internet. While TinyMUD wasn't the first MUD, its easy interface and portability to Unix systems was the origin of its success and it caused an explosion of MUD popularity that has not yet waned (Burka, 1993).

These systems were technically very simple and did not involve the use of any special devices as for immersive VR. The game was played in several separated virtual spaces, or *rooms*, so that people and objects in one room could not directly interact with people and objects in others. All the interactions took place through text

since neither pictures nor sounds were supported. The transfer of information to the network was handled through TCP sockets and at the beginning of the session each user would take control of a computerized avatar to interact with others. The participants could walk around, chat with other characters, explore areas, solve puzzles and even create their own rooms, descriptions and items.

The first environments were initially simply textually described, but subsequently they became more graphically realistic.

There are many different types of MUDs (Burka, 1993), and each specializes in a different sort of user-interaction and scope. The *Tiny-* and *Teeny-* family of MUDs are usually more socially oriented; the players in those MUDs tend to gather, chat and meet friends to discuss all kinds of topics (Burka, 1993). The *LP-* family of MUDs, including *Diku* and *AberMUD* are usually based on role-playing adventure games (Burka, 1993). Many others have appeared such as *MOOs* and *UnterMUDs* and each has its own unique style (Burka, 1993).

Elizabeth Reid describes MUD systems as the first form of Collaborative VR: "A MUD program is, in essence, a set of tools that can be used to create a sociocultural environment. It is this that sets MUDs apart from other textually based, computer-mediated communication tools. The latter merely provide an interface that separates what a person types from that of another and so allows a form of written conversation. MUDs, by contrast, allow the depiction of a physical environment that can be laden with cultural and communicative meaning. They allow imagination and creativity to furnish the void of cyberspace with socially significant indicators. It is this that makes a MUD system a form of virtual reality" (Reid, 1995, p. 167).

A MUD creates a non-graphical virtual environment that exists not only in databases or in the computer networks supporting the systems but also in the imagination of all the participants. In short a "MUD program serves to actualise what is imagined in ways that can be communicated to or retrieved by other users" (Reid, 1995, p. 183).

3.4.3 Multiple-Users Navigation Systems

This category of CVE systems includes all the systems that allow multiple users to navigate virtual environments and to communicate with each other using several communication channels. In this category virtual environments are not text-based, as in MUD systems, but are displayed using a variety of VR visualization devices.

This is the most popular CVE category and very many systems have been built for a variety of purposes. A sub-categorization that divides these systems into seven groups follows:

- Cyber-communities: for entertainment purposes or educational and social research
- Virtual cities: for entertainment and education
- CVEs for design: to explore design solutions interactively and for educational purposes
- CVEs implementing realistic avatars: to enhance serious communication or for entertainment
- CVEs for education: used primarily as didactic tools and for virtual storytelling
- CVEs as artistic expression: shared VEs are unconventional spaces that can generate creative fillings and unusual sensations in their visitors
- Multi-disciplinary CVE systems: often broad projects that can offer the opportunity to investigate several aspects of CVEs involving different disciplines.

In the following pages examples of different categories will be provided and the relevant collaborative issues will be discussed.

3.4.3.1 Cyber-Communities: the HABITAT example

Lucasfilm's Habitat (Morningstar et al., 1990) was a distributed virtual environment capable of connecting thousands of users in a single shared cyberspace. It was one of the first attempts to create very large-scale commercial multi-user VE for entertainment purposes and it was inspired by the cyber culture of the 1980s, in particular by Vernor Vinge's novel, *True Name* (Vernor, 1981).

Habitat was built on an ordinary commercial online service and was accessible through ordinary PCs.

Habitat gave a real-time animated view to the online simulated world in which users could communicate and interact in a virtual community.

The initial release of the system was for Commodore 64 home computers. This version of the system was able to display the virtual world and the embodiments of the participants in real time. The virtual scenario consisted of a number of objects such as houses and trees, and the animated avatars, who were often of human appearance were commanded using a joystick. Avatars were able to navigate the scene and also to interact with the objects by choosing and manipulating them.

Communication among the participants was accomplished by using textual chat-lines and the typed text was displayed above the users' avatars in a cartoon style "word balloon".

The architecture of Habitat was based on a large number of locations, called *regions,* which were interconnected to create a continuous virtual space. Using doorways passages or moving to the edge of the screen were among the means to access various linked regions.

In each region avatars could interact with a set of objects which could perform certain functions (See Figure 3.5).

Object Class	Function
ATM	Automatic Token Machine; access to an Avatar's bank account
Avatar	Represents the player in the Habitat world
Bag, Box	Containers in which things may be carried
Book	Document for Avatars to read (e.g , the daily newspaper)
Bureaucrat-in-a-box	Communication with system operators
Change-o-matic	Device to change Avatar gender
Chest, Safe	Containers in which things can be stored
Club, Gun, Knife	Various weapons
Compass	Points direction to West Pole
Door	Passage from one region to another; can be locked
Drugs	Various types; changes Avatar body state, e.g., cure wounds
Elevator	Transportation from one floor of a tall building to another
Flashlight	Provides light in dark places
Fountain	Scenic highlight; provides communication to system designers
Game piece	Enables various board games: backgammon, checkers, chess, etc.
Garbage can	Disposes of unwanted objects
Glue	System building tool; attaches objects together
Ground, Sky	The underpinnings of the world
Head	An Avatar's head; comes in many styles; for customization
Key	Unlocks doors and other containers
Knick-knack	Generic inert object, for decorative purposes
Magic wand	Various types, can do almost anything
Paper	For writing notes, making maps, etc.; used in mail system
Pawn machine	Buys back previously purchased objects
Plant, Rock, Tree	Generic scenic objects
Region	The foundation of reality
Sensor	Various types, detects otherwise invisible conditions in the world
Sign	Allows attachment of text to other objects
Stun gun	Non-lethal weapon
Teleport booth	Means of quick long-distance transport; analogous to phone booth
Tokens	Habitat money
Vendroid	Vending machine; sells things

Figure 3.5: Objects and related functions in Habitat (Morningstar et al., 1990)

Habitat was very close to the multi-player games that are now proliferating throughout the Internet. It was one of the first attempts to involve a large number of users, around 20,000, and therefore it gave the researchers the opportunities to study cyber-community behaviour and observe problems in communication which has been mediated by computers.

3.4.3.2 Virtual Cities: the VR-Glasgow example

The Glasgow Directory (Maver et al., 2000; Ennis et al., 2001; The Glasgow Directory, 2002) and its multi-user version, VR-Glasgow (VR-Glasgow, 2002) were launched in 1999 by the ABACUS group at the University of Strathclyde in Glasgow. This web site allows the user to explore a three-dimensional model of Glasgow, to meet multiple participants in the virtual city and to retrieve information through many links to various information sources. These links can be accessed either by using traditional lists, tables and search engines or while still navigating the

25 km^2-wide VRML model of the city (Peng et al., 2002) directly by clicking information categories in a Menu (See Figure 3.6). Once a category of interest has been chosen, the location of related buildings or sites are directly visualised in the virtual city, and links to more detailed information are provided. The system can provide the user with information from:

- A multimedia database of General Tourist Information
- A multimedia database of Glasgow Architecture
- An alphanumerical database of property addresses and street names
- External Web sites

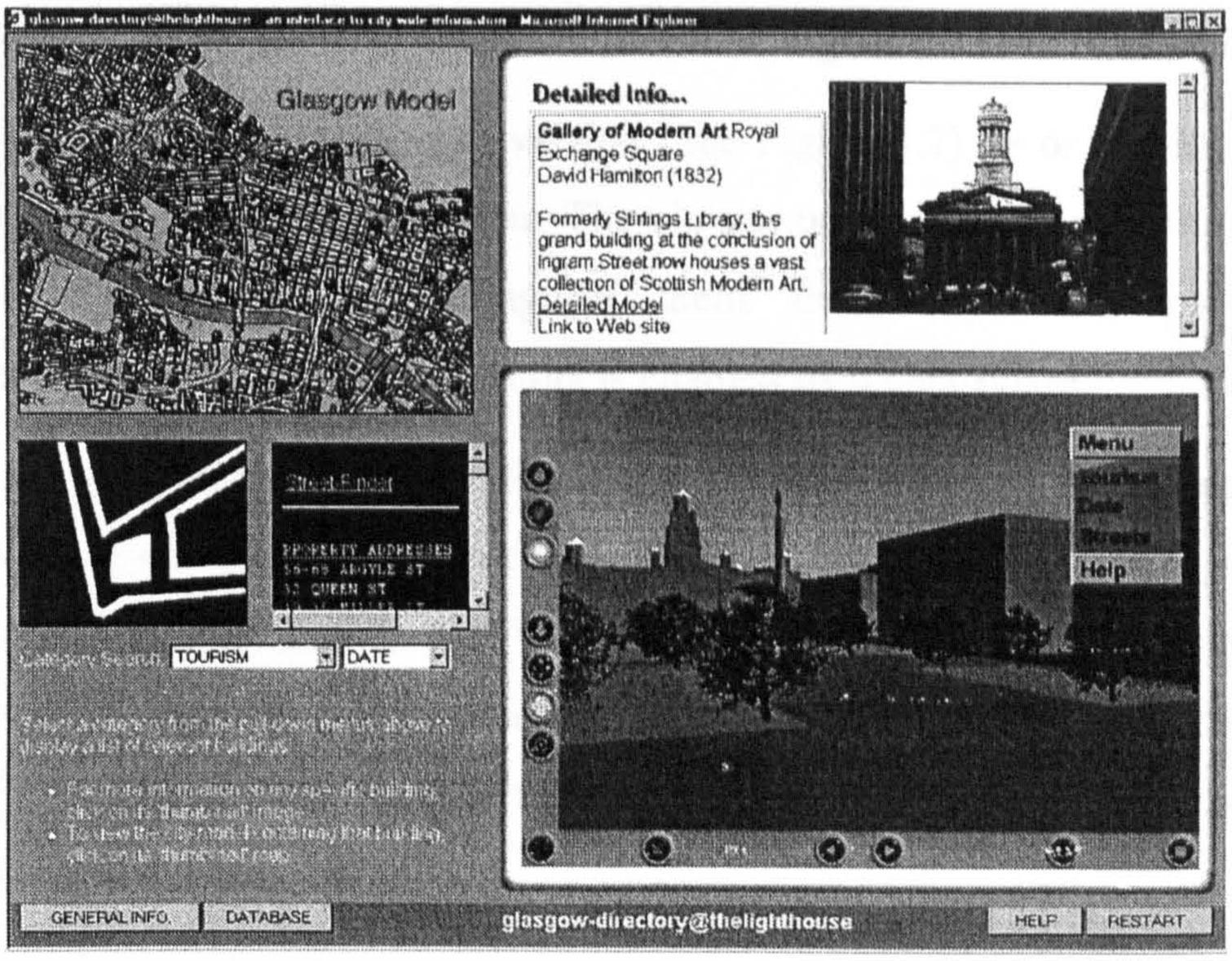


Figure 3.6: The Glasgow Directory, a screenshot from the web site (The Glasgow Directory, 2002)

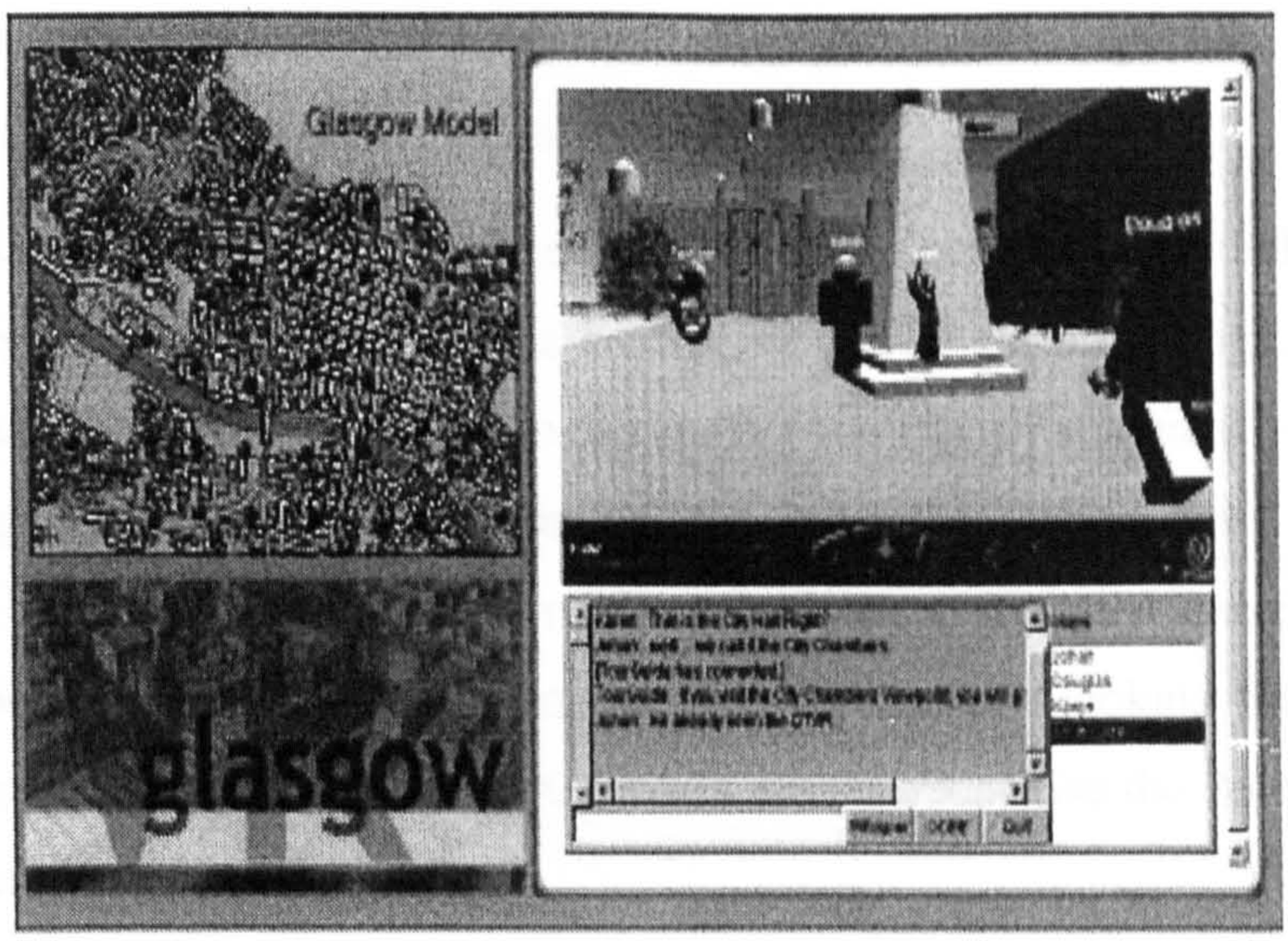


Figure 3.7: The multi-user interface to VR-Glasgow (Maver et al., 2000)

Once the user logs in the multi-user version (See Figure 3.7) he or she is asked to insert a name and to choose an avatar. The chosen name is displayed close to the avatar's head in order to distinguish them from any other participant. Communication among the virtual visitors is enabled by a chat-board.

In terms of used technology, the shared virtual environment is based on the VNET client/server package, a Java™ application that allows multiple users to communicate using chat boards. The VRML virtual world is downloaded onto each user's computer once they are connected to the VNET server. This reduces the amount of data transmission to only the coordinates of the various users positions and the text for the chat board.

3.4.3.3 CVEs for Design: CASA, CALVIN and LCD-VR Systems

CASA (Leigh et al., 1995) is the acronym for Computer Augmentation for Smart Architectonics and it was developed at the Electronic Visualization Laboratory (EVL), at the University of Illinois, Chicago, where the CAVE was also developed (Cruz-Neira et al., 1993).

It is a system that was designed to explore virtual architectural spaces using the immersive VR devices that were available at EVL. In addition to the interaction provided by normal use of the CAVE consisting of moving a tracked wand fitted with three push button inputs and a pressure-sensitive joy-pad, the CASA system

allowed three more means for interaction; through voice-recognition, continuous-stroke gesture recognition and a virtual visor.

This voice recognition module was an integration of commercially available packages for speech recognition. The continuous-stroke gesture recognition feature allowed the user to train the system to recognise continuous movements of the wand. By pressing one of the three buttons on the wand the system starts the gesture recognition procedure until the user releases the button. Once the movement of the wand has been completed, the system examines its database looking for the gesture that was just executed. For example CASA is able to recognise the 26 characters of the alphabet (See Figure 3.9).

Finally the virtual visor is a head-up display (HUD) on which information on instrumentation is displayed. HUD technology has been extensively investigated for military applications (Fisher, 1980) and it provides a successful interface to interact with the virtual world. When the visor is activated a targeting cursor appears in front of the user that can be moved through a gaze tracking system. When the user moves the cursor he can select a visor item by pressing a button on the wand.

CASA allows including a number of sensors in the virtual building and these can be triggered once the user approaches them. The "smart" objects are programmable, via a scripting language, in order to control for instance the illumination of a room or to perform other actions.

Within the CASA framework a CAVE-to-CAVE project has been carried out with the goal of linking multiple networked participants exploring the same CASA space. Avatars in this case consist of a separate head, body and hand, whose positions and gestures are transmitted to the other participants.

The networked version of CASA was implemented using a communications library developed at EVL, called SpiffNet that was coded in C++. The underlying client/server architecture has a central server that maintains a listing of all the clients connected and it broadcasts data according to data rates compatible with the various clients thus preventing slow clients from receiving unmanageable superabundant data.

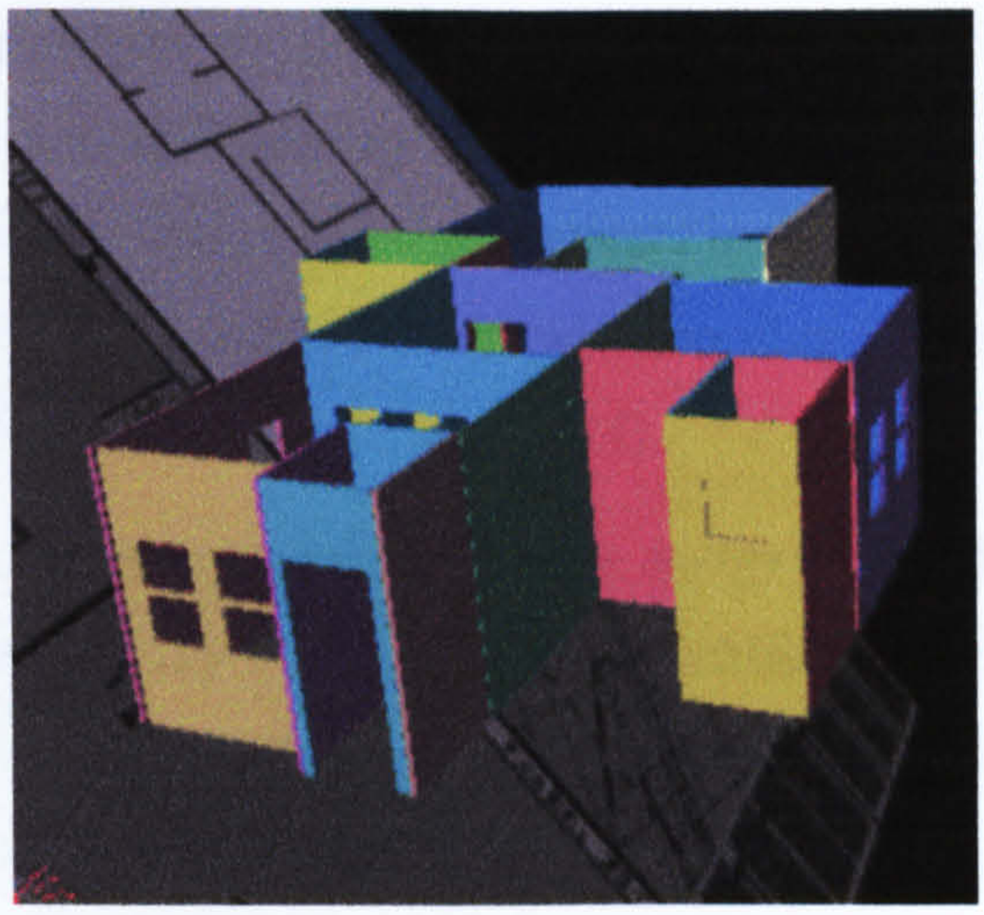


Figure 3.8: An image of a 3D space navigated using CASA (Leigh et al., 1995)

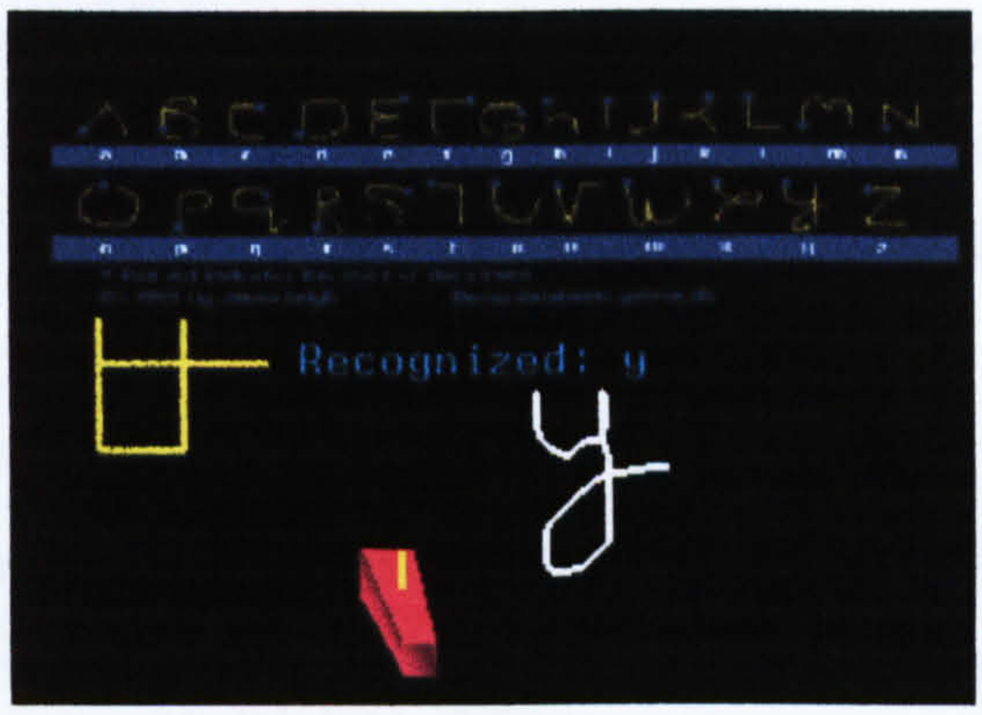


Figure 3.9: Gesture recognition in CASA (Leigh et al., 1995)

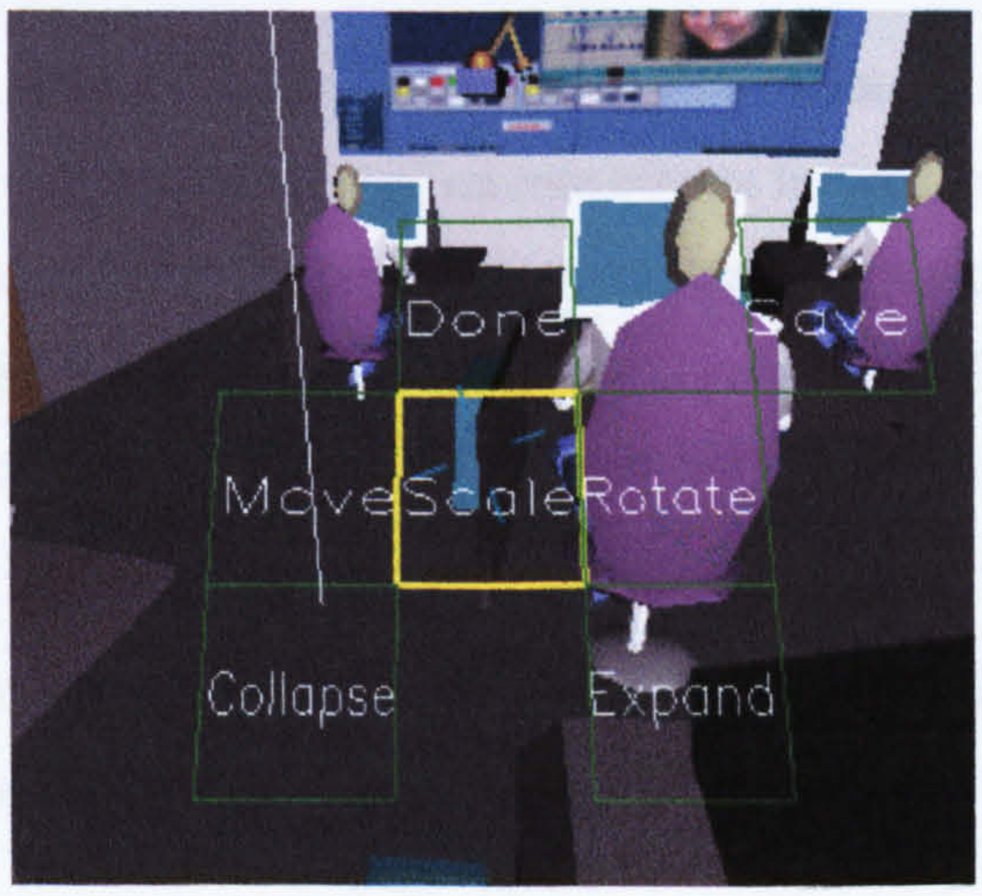


Figure 3.10: Virtual visor of CASA (Leigh et al., 1995)

Another example of a CVE for design is CALVIN (Leigh et al., 1996), which stands for Collaborative Architectural Layout Via Immersive Navigation. It is a

prototype system developed on the CASA framework at EVL that allows multi-camera perspectives and multiple participants to concurrently navigate in a shared architectural environment. Participants can interact in the VE using two perspectives: the global view and the miniaturised view, according to the task they have to accomplish in the virtual scene (See Figure 3.11).

A selection of avatars is provided whose design was studied carefully in order to obtain the highest visibility when placed in the virtual world (See Figure 3.12). In CALVIN avatars are capable of gesture just as they are in CASA thus enhancing communication (See CASA in this section).

Users are provided with LCD shutter glasses to give a view in stereo and with a 3-button wand. Both devices are attached to a magnetic tracker that streams the user's head and hand positions to the network. CALVIN was originally conceived to work in a CAVE virtual environment but it also supports other VR configurations such as ImmersaDesk, BOOM, fish-tank VR systems and simple graphic workstations.

In terms of the users' interface the system allows two means for this interaction: the virtual visor (See Figure 3.13) and a speech recognition module. The virtual visor is the previously mentioned HUD inherited from the CASA system.

As an alternative to the virtual visor CALVIN also incorporates a speech recognition module that enables voice commands to be used.

In terms of network architecture and collaboration, CALVIN is built on a client-server model with the number of remote clients limited only by bandwidth and latency. A central repository assures data consistency throughout the network by managing the transfer of information related to the collection of objects placed in the VE, the avatars present in the virtual scene and the collection of scene description files.

CALVIN's implementation relies on the use of Open Inventor as a graphics library and it is entirely coded in C++.

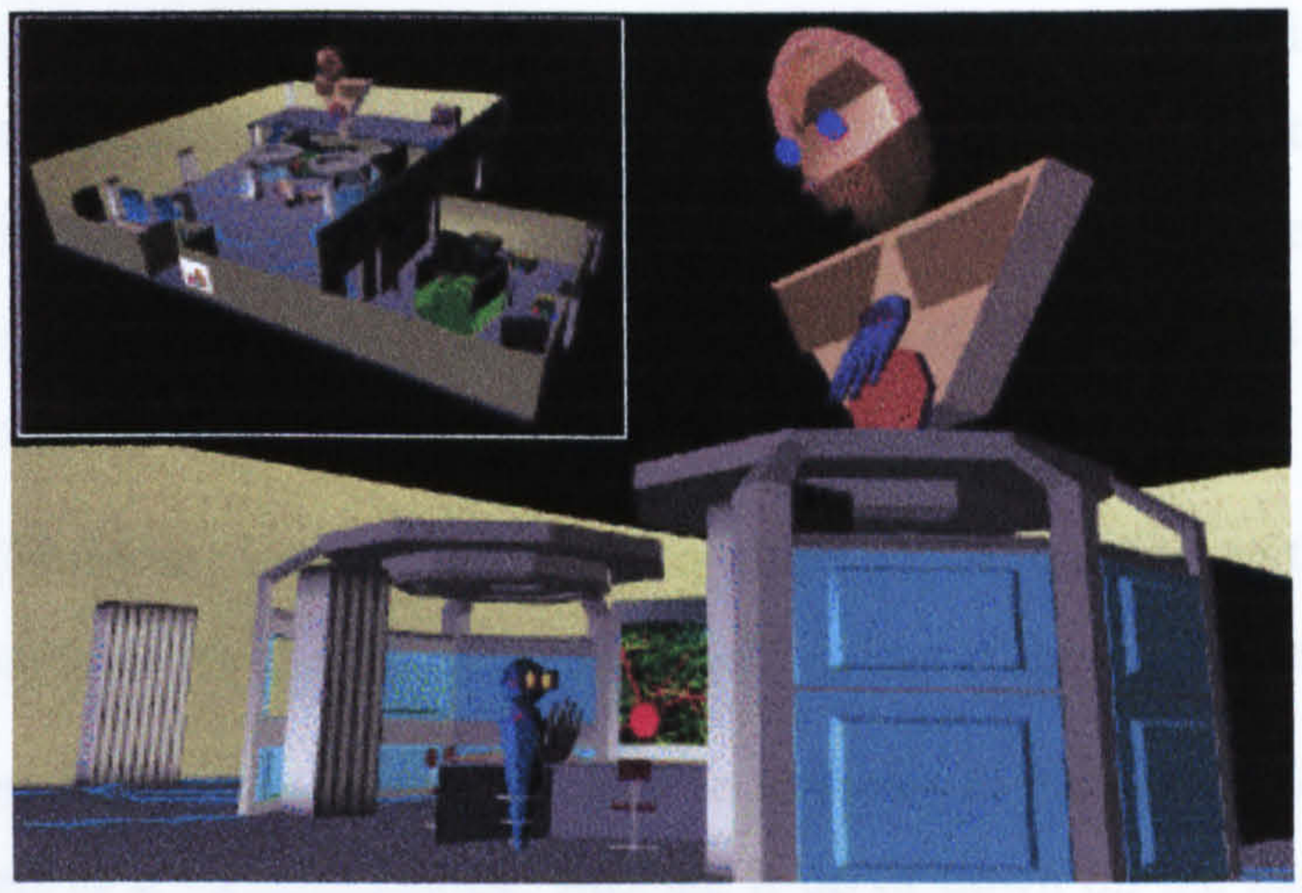


Figure 3.11: Participants can switch from a global view to miniaturised view (Leigh et al., 1997)

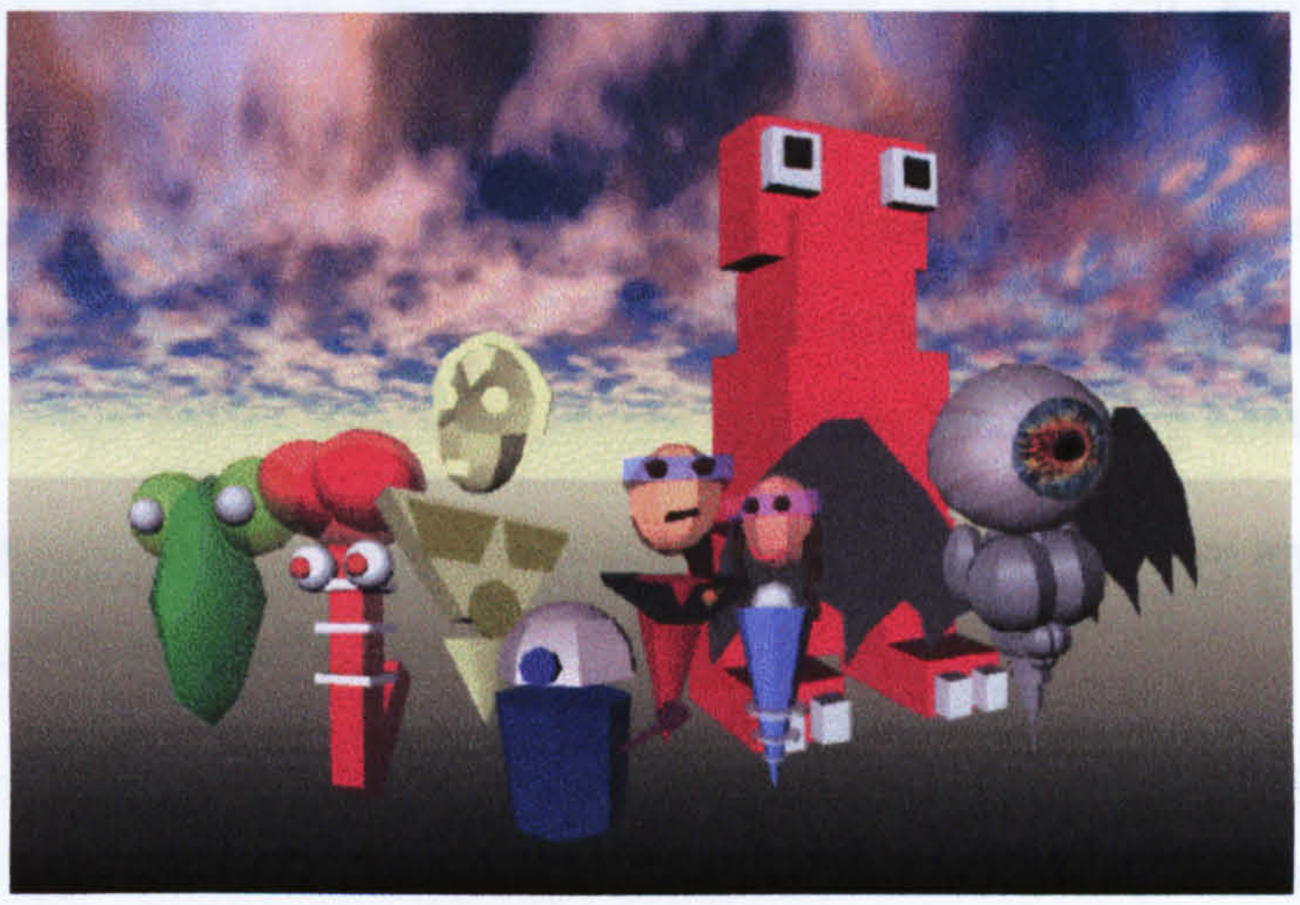


Figure 3.12: A selection of CALVIN's avatars (Leigh et al., 1996)

Figure 3.13: A view of CALVIN's virtual visor (Leigh et al., 1996)

The last example of CVE for design is LCD-VR (Low Cost Distributed VR) system, a stand alone or multi user system which can visualize and interact with 3D models over the net. Figure 3.14 shows the graphic user interface of the system and a detail view of the control panel through which most of the important functions are accessed. These include:

- *Network functions:* icons in this section allow the connection to the server, the disconnection of the system and showing the status of the connection.
- *Movement:* this section allows the user to navigate in the 3D world. The control is designed as a virtual joystick. Moving it up or down on the panel moves the user forward or backwards. Left or right moves will turn the user left or right.
- *View:* this section deals with the camera's viewpoint within the virtual window. It allows the rotating of the camera viewpoint, down, left and right at the convenience of the user.
- *VR functions:* icons in this section affect functions relating to the virtual environment. It is possible to create "virtual notes" regarding future actions, to turn on and off the VR environment and to access the viewing controls.
- *General functions:* this section provides the ability to open, save and clear the 3D models and to open a chat window.

In terms of software implementation and architecture, LCD-VR consists of two main parts: the VR-Server that displays and manages three-dimensional data, and the *main program* that establishes network connections and maintains a database and a geometric description of the model. The system is entirely coded in C++ and the management and visualization of 3D models in VR-Server is achieved using the Renderware library.

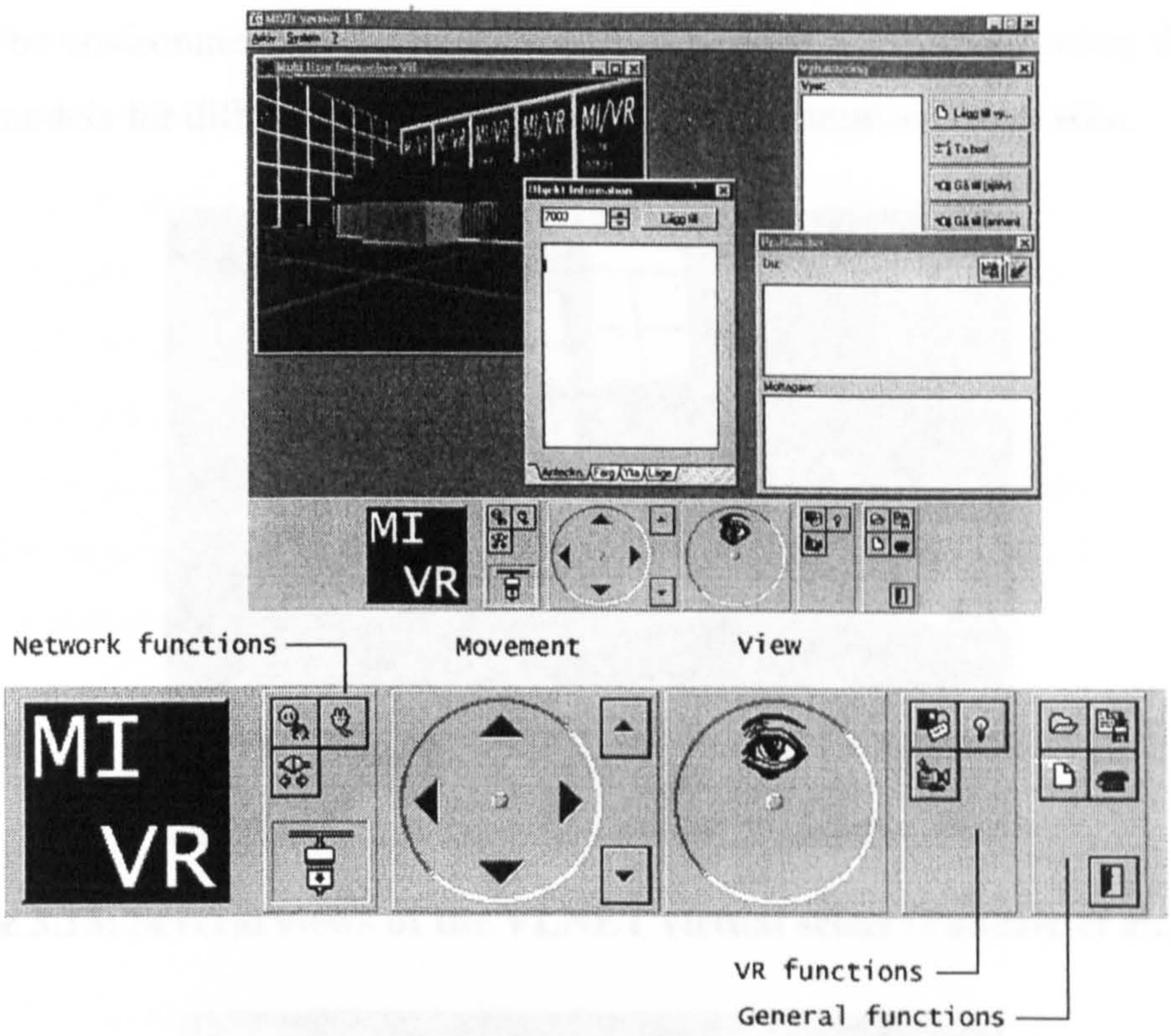


Figure 3.14: LCD-VR interface and a close up view of the control panel (Lindemann, 1996)

3.4.3.4 CVEs Implementing Realistic Avatars: the VLNET System

The Virtual Life Network (VLNET) (Capin et al., 1998) is a CVE system incorporating highly realistic virtual human representations and intelligent avatars. VLNET supports a shared virtual environment and allows multiple users to interact with the environment and the other participants. Users are represented in the VE via 3D virtual actors resembling human beings, in order to enhance the virtual experience and the sense of presence in the VE. The human form of representation also allows facial and gestural communication between the users.

The system incorporates different media and integrates them in a single 3D world. In order to enhance communication participants are provided with control of body postures, facial expressions using real-time video analyses or by accessing a menu of emotions. They also have the possibility of projecting real-time video onto any 3D object in the virtual scene.

The environment works as a general-purpose framework allowing the use of various models for different applications and future customised expansion.

Figure 3.15: Several views of the VLNET virtual scene (Pandzic et al., 1995)

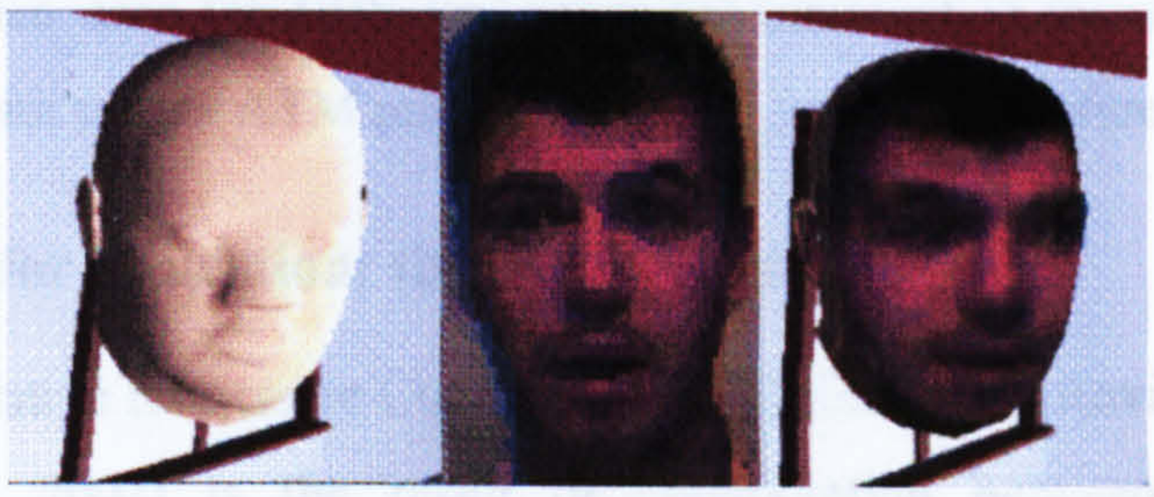


Figure 3.16: Real-time video mapped onto a 3D object (Pandzic et al., 1995)

In terms of object manipulation, objects are classified into two groups, fixed, (walls), or free (pieces of furniture). Only free objects can be chosen, moved and edited.

A great deal of attention was given to the implementation of the behavioural abilities of the avatars or "virtual actors" (Pandzic et al., 1995). Walking behaviour is based on the human walking and gesture model that includes customisable kinematic parameters depending on the user that moves the virtual actor. By setting the speed, orientation and the user's personification parameters, the walking module generates the movement of the avatar by articulating its virtual body.

Grasping of objects ability was also implemented in order to achieve realistic looking motion by the virtual actors. The user indicates the object they have chosen

and the grasping module automatically identifies the appropriate movement that the virtual actor has to perform suited to the type of object.

Facial expression and therefore interaction was achieved in the virtual scene by the mapping of the users' face onto the virtual actors' heads. In order to transmit the moving image of the users' face, captured by a video camera, an analysis of the real-time video stream is completed: once the image of the face is isolated, it is first compressed, using the SGI Compression Library MVC1 compression algorithm, and it is then sent to the other participants screens. At the receiving end the streamed image is mapped onto the virtual actor's head as a texture map.

VLNET allows asynchronous collaboration among the participants through support of action such as the change of position and orientation of the avatars, as well as the grasping and releasing of objects. Once a change in the VE is made it is broadcasted to all the users.

The system uses the IRIS Performer environment to manage the real-time graphic application and an ATM pilot network for the video-conference streaming.

3.4.3.5 CVEs for Education: the NICE example

The Narrative Immersive Constructionist/Collaborative Environments project (NICE) (NICE, 2002) is the result of collaboration between the Interactive Computing Environments laboratory (ICE) and the Electronic Visualization Laboratory (EVL) at the University of Illinois in Chicago.

NICE (Johnson et al., 1998 and 2001) is a collaborative environment built for educational purposes, based on two previous systems implemented at EVL: CALVIN and the Graphical StoryWriter. CALVIN (See Section 3.4.3.3) is a networked collaborative environment for designing architectural spaces. The Graphical Storywriter is a shared workspace where young children can develop and create structurally complete stories. In 1996 the combination of these two applications was used in the development of the NICE prototype, which was specifically designed to help children understand the basic concepts of plant growth and gardening.

The virtual scenario here is of an island where the children are invited to construct micro ecosystems. The children decide where to build their garden,

rainforest or grapevine, depending on the different types of soil, altitude and weather conditions etc. Moreover on the island there are a variety of exciting spaces that invite exploration such as a dormant volcano and a labyrinth of caverns under the island.

Children are represented in the scene through avatars with gesture capabilities inherited from the CALVIN system, (See Section 3.4.3.3). They can collaboratively plant, grow and pick vegetables and flowers. The children's tasks are also to ensure that the plants have sufficient water, sunlight, and space to grow and they must also avoid hungry animals eating them.

Figure 3.17: The virtual garden of the NICE VE (NICE, 2002a)

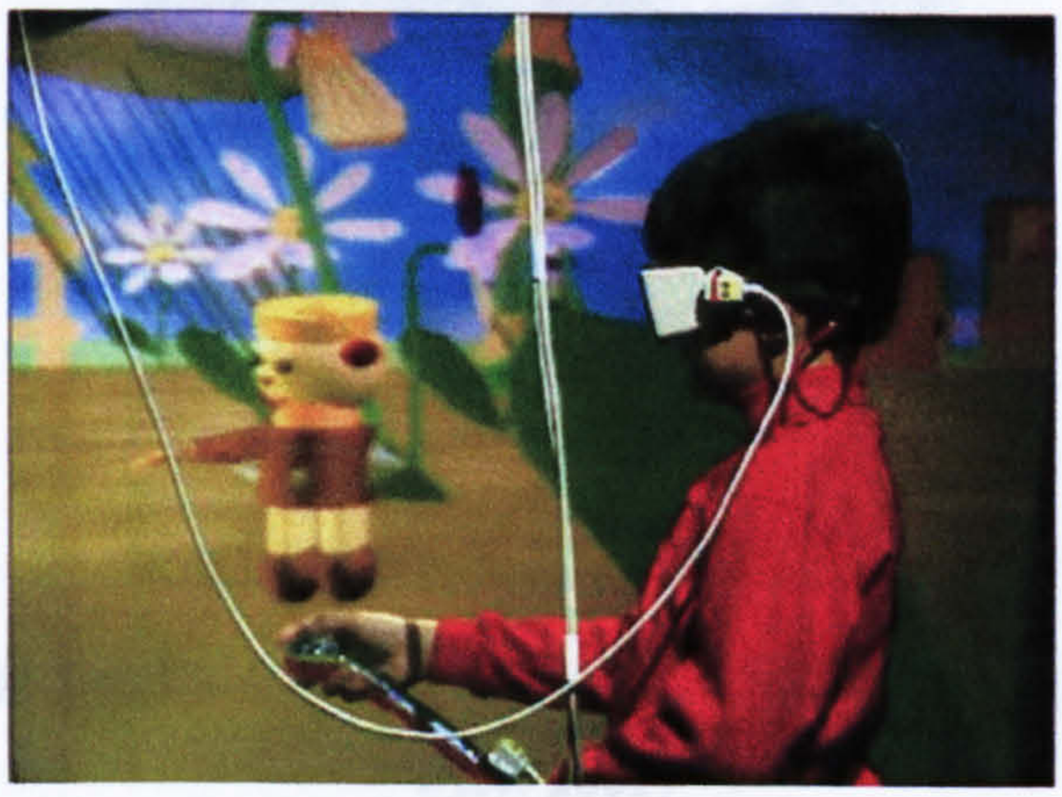


Figure 3.18: A boy using NICE in the CAVE (Johnson et al., 1998)

Children can also change their avatars size and transform themselves into mice in order to walk under the flowers and see the roots of the plants etc.

NICE is developed for use in the CAVE and communication among the participants is by using microphones connected to a conference call system. Unlike the CALVIN system, (See Section 3.4.3.3), it is a persistent environment, which means that the virtual environment continues to exist even if there are no users connected, for example the plants keep growing and the autonomous creatures that inhabit the island will stay active.

NICE was shown in several SIGGRAPH conferences since 1996 where it was a successful socialising space and a useful test bed for tele-immersion.

3.4.3.6 CVEs as Artistic Expression: the Distributed Legible City

The Distributed Legible City (Crabtree et al., 2000 and West et al., 2001) is an example of a shared electronic artistic space. People are invited to cycle into a virtual city and communicate using audio links.

The Distributed Legible City (Distributed Legible City, 2002) is based on The Legible City, an original art piece created by Prof. Jeffrey Shaw at the Center for Art and Media (ZKM) in Karlsruhe, Germany.

The Legible City consisted of three models of the cities of Amsterdam, Manhattan and Karlsruhe where the facades of the buildings were created from textual narratives that the viewer could read while exploring (See Figure 3.19).

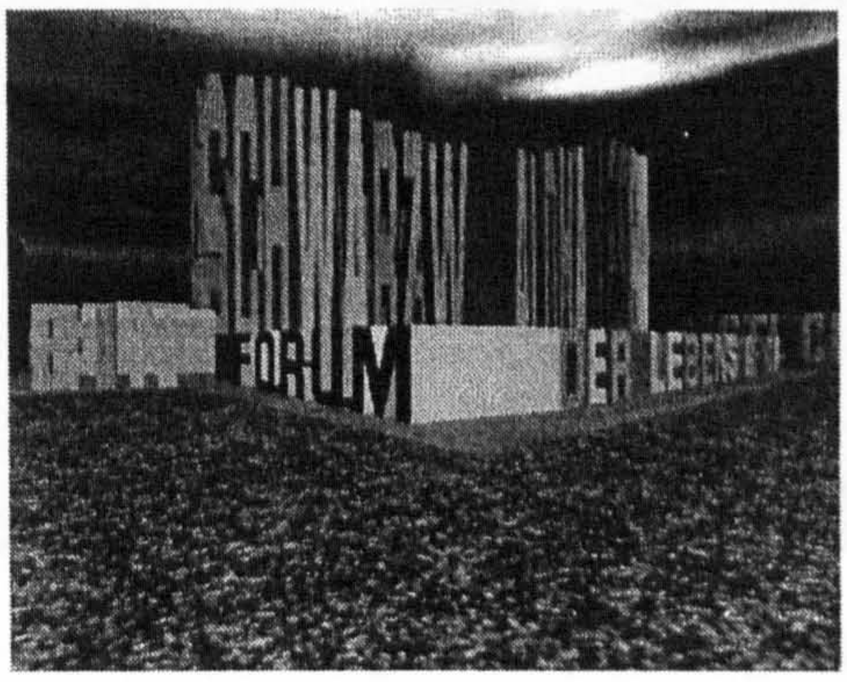


Figure 3.19: The Legible City implemented in Maverik (Distributed Legible City, 2002)

Participants could navigate the virtual scenario by riding a modified exercise bicycle placed in front of a large multi projector display system. The original implementation by Gideon May was in 1990 when it was run by SGI workstations.

The next step in its development was the implementation and optimisation of the Legible City in Maverik, a publicly available VR system that had been under development by the Advanced Interfaces Group (AIG) since 1995.

Maverik includes the Level-of Detail (LOD) function to lighten the model and it has three levels of accuracy in the text rendering depending on its distance from the user: when text was close to the viewer it was rendered in full detail, once the distance between them increased it was rendered as a solid box and eventually as the distance increased it was removed completely from the scene.

Fogging was employed to mask the transition between these stages. Texture maps and an overview map were also added to the original code. The last version, which was called the Distributed Legible City, used the Deva system, also developed at AIG to add multi-user network capabilities to the Maverik virtual environment. The code written for Maverik was wrapped and used within Deva to achieve a distributed version of the system (See Figure 3.20 and Figure 3.21).

Figure 3.20: The Distributed Legible City during a multi-user session (Distributed Legible City, 2002)

The Distributed Legible City features animated avatars in the form of cyclists who pedal and steer as in the real world, and who have communication capabilities through an audio link and an overview map accessible from a button on the user's cycle's handlebars.

Several installations of the system were shown at various events, including the ESPRIT IST98 conference and the Surrogate1 exhibition, where three bicycles placed in ZKM were connected to different museum halls.

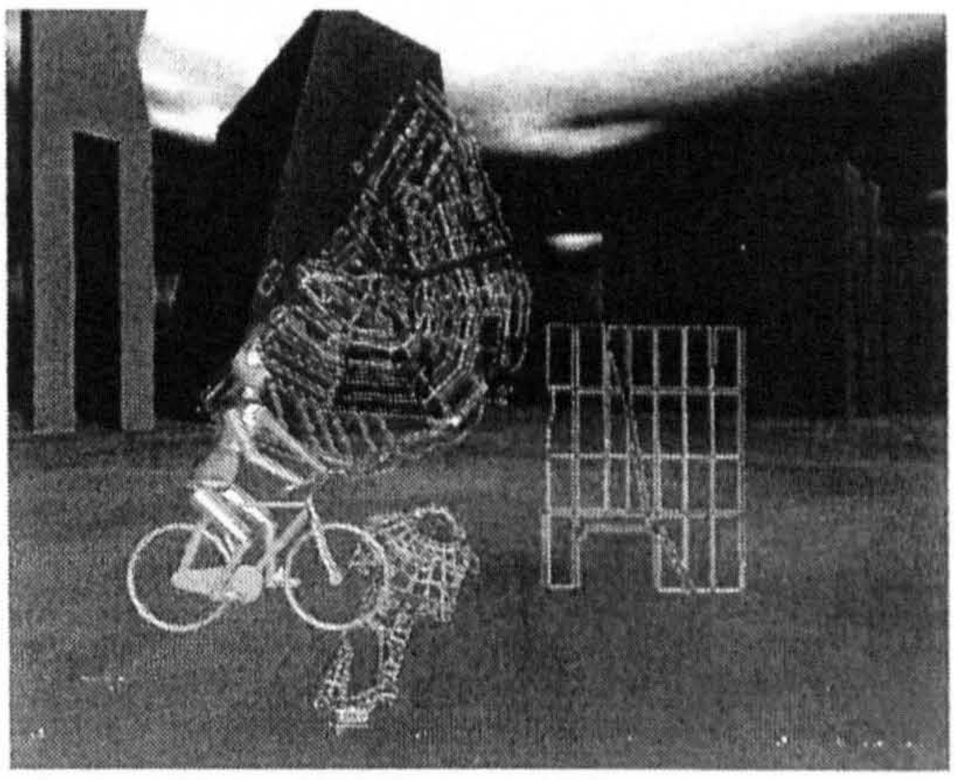


Figure 3.21: A screenshot of the system with the virtual cyclist avatar and the overview map (Distributed Legible City, 2002)

3.4.3.7 Multi-Disciplinary CVE Systems: the COVEN Project

The COVEN (Collaborative Virtual Environments) (COVEN, 2002) system was the outcome of a four-year (1995-1999) project funded by the European Community. It was a broad based project under the co-ordination of Thomson-CSF and it included many partners from several different countries. The research focus was to develop a computational service for teleworking and virtual presence.

The project explored the development of several virtual scenarios as a test for using a variety of communication tools, techniques for ground navigation, speech interactions and real-time animated virtual human figures. Among these scenarios were: a Virtual Conferencing scenario, a Business Game scenario, an Interior Arrangement scenario, a Travel Rehearsal scenario and the City of London.

The *Virtual Conferencing scenario* is the generic scenario that was used to provide the set of supporting generic functions that would be used in the other scenarios.

The *Business Game scenario* (See Figure 3.22) allows the exploring of the communication needs in negotiation situations.

The *Interior Arrangement scenario* (See Figure 3.23) provides a collaborative 3D design task scenario where 3D objects can be changed and manipulated collaboratively by the users, and actual speech ensures communication among the participants.

The *Travel Rehearsal scenario* (See Figure 3.24) is a large virtual environment of parts of the city of London and surroundings areas, and it was built mainly to benchmark future network performances, when dealing with very large VEs.

Finally the *citizen-oriented scenario* (See Figure 3.25) is a large Virtual Travel Agency scenario where users are able:

1. To access a large quantity of data in several formats such as text files, images and audio
2. To experience virtual visits to archaeological sites
3. To access on-line information through the WWW
4. To meet other people with the same holiday interests to gather information together and eventually to book trips together.

Within this a number of separate virtual environments are linked each has its own purpose and the users are able to freely move from one of these environments to another.

In terms of implementation the entire COVEN project extends and is based on the Distributed Interactive Virtual Environment (DIVE) system (Normand, 1996 and Normand, 1998) (See Section 3.4.4.2). This was an internet-based multi-user VR system that allowed the user to navigate in virtual scenarios and to interact freely with other participants and applications. Its first version appeared in 1991.

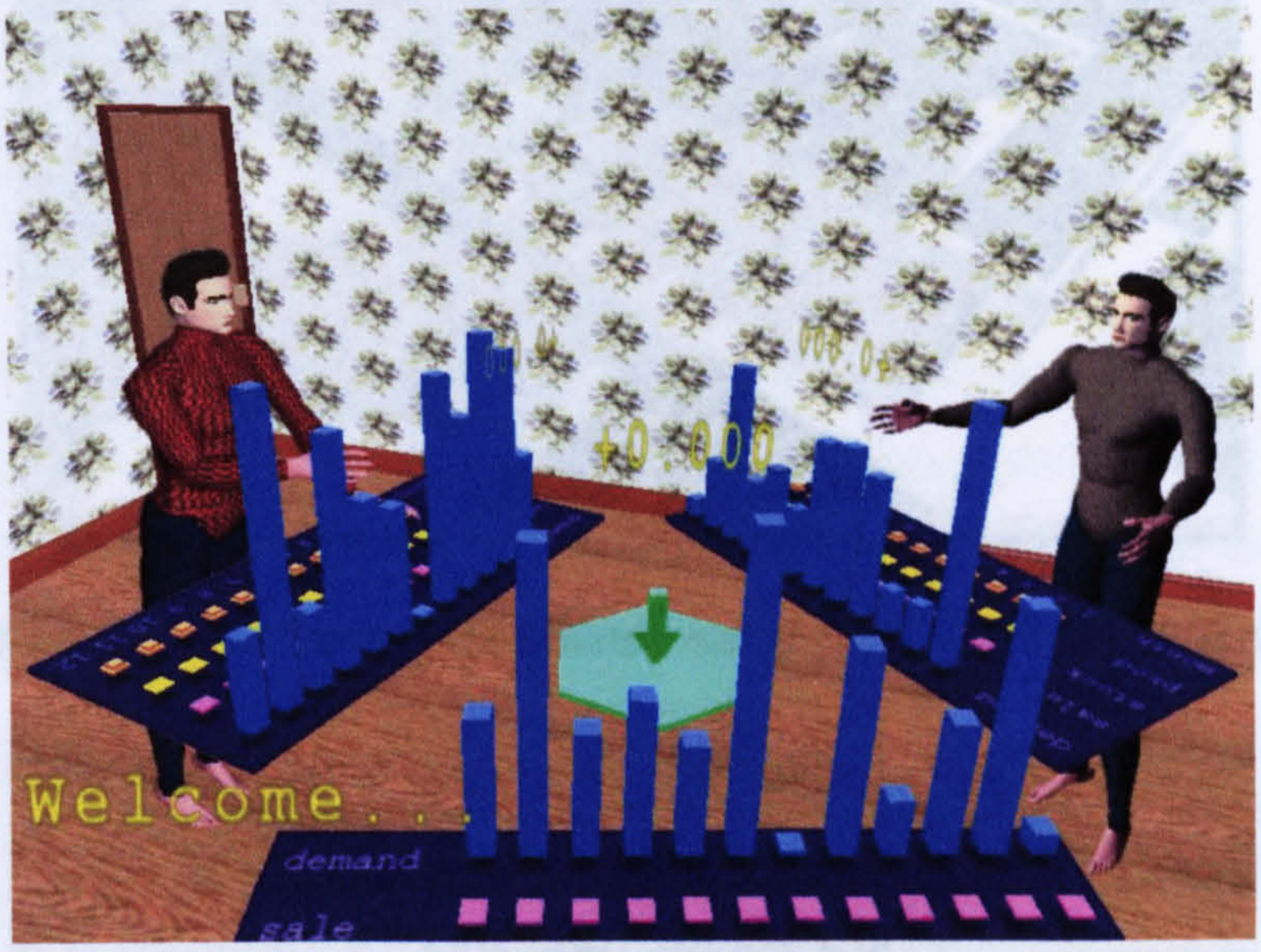


Figure 3.22: The Business Game scenario (COVEN, 2002a)

Figure 3.23: The Interior Arrangement scenario (COVEN, 2002b)

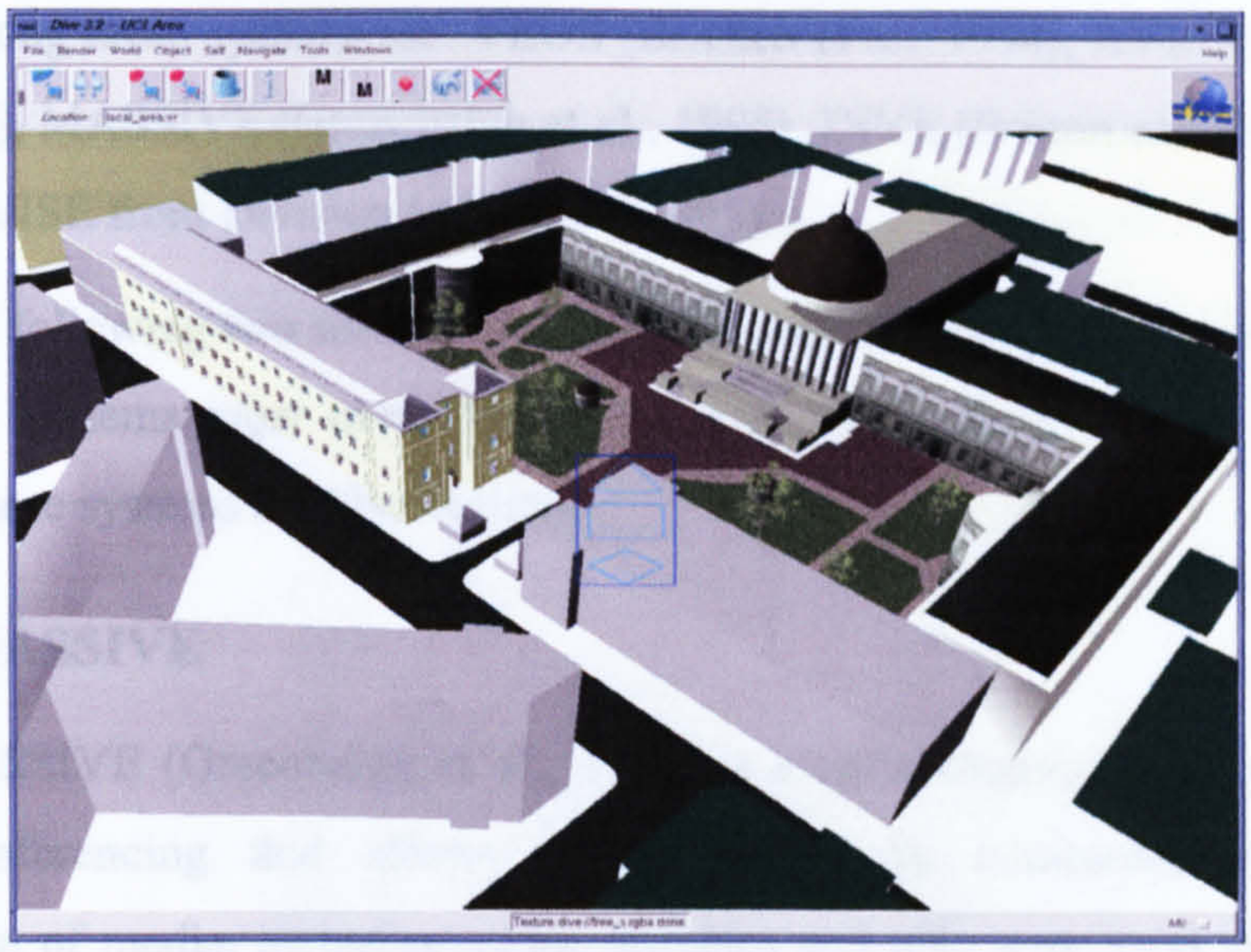


Figure 3.24: Travel Rehearsal scenario (COVEN, 2002c)

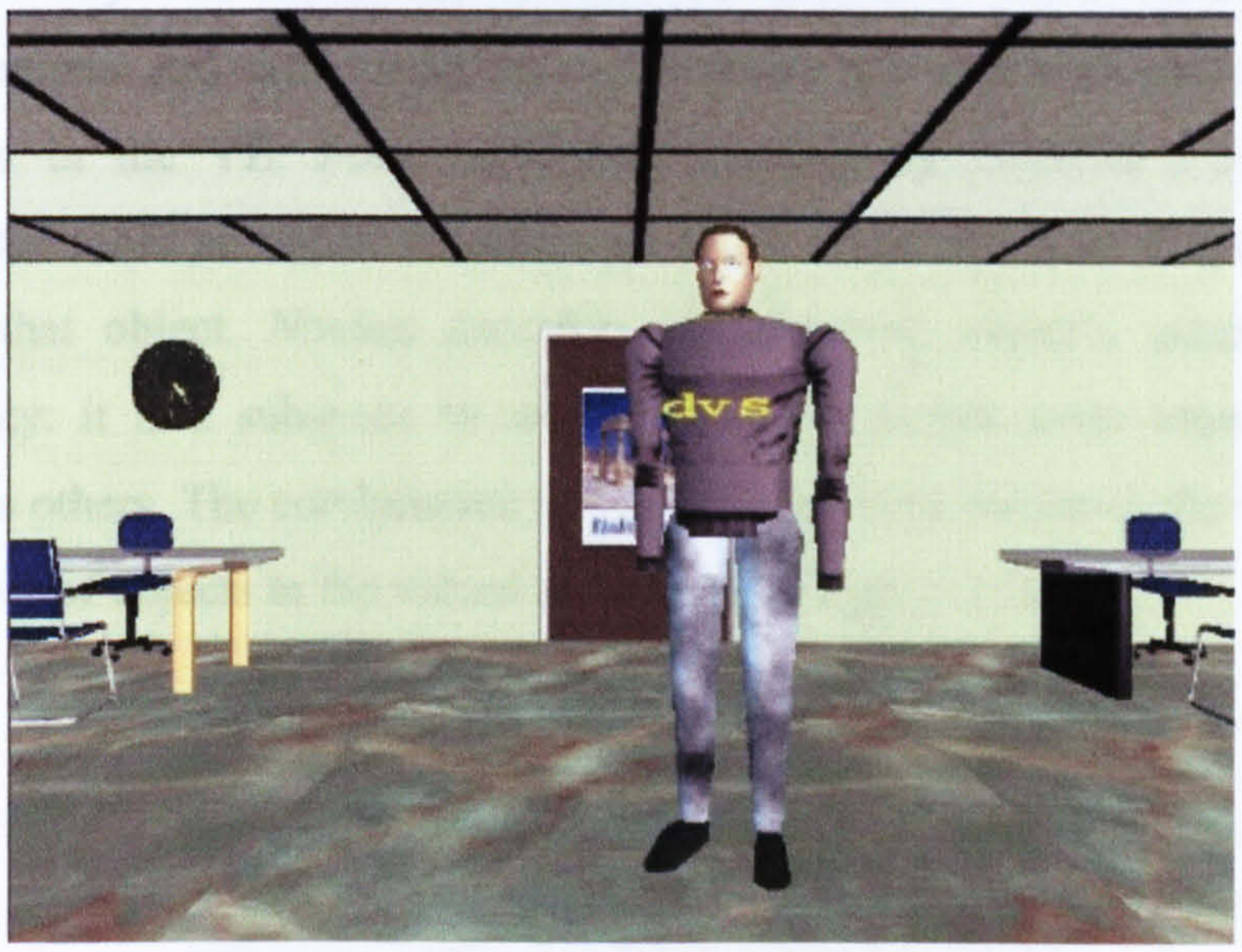


Figure 3.25: The Virtual Travel Agency of the citizen-oriented scenario (COVEN, 2002d)

3.4.4 Platforms for Implementing VR Shared Applications

Several systems have been implemented by various institutions, companies and universities to provide customisable base-line platforms for building collaborative virtual environments. These systems are usually implemented on complex and well-optimised network architectures.

Among these systems are: VEOS (Bricken et al., 1994), AVIARY (Snowdon et al., 1994), MASSIVE (Greenhalgh et al., 1995), DIVE (Frécon and Stenius, 1998) and dVS/dVISE from Division Ltd (Ghee, 1995).

The following two sections will provide details of the software architecture of two of the systems most used, MASSIVE and DIVE. Several CVE applications based on these systems are also outlined.

3.4.4.1 MASSIVE

MASSIVE (Greenhalgh et al., 1995) is a collaborative virtual environment for teleconferencing that allows multiple users to communicate through a combination of media, including audio, graphics and text, over local and wide area networks. MASSIVE uses a complex spatial model of interaction, which is based on the concepts of *focus*, *nimbus* and *aura*, which depend on users' relative positions and orientations and will amplifies or weakens the user's awareness of other participants in the VE. *Focus* describes spatially the observer's allocation of attention: the more an object is within an user's focus the more that user will be aware of that object. *Nimbus* describes the observed object's manifestation or observability: it is a subspace in which an object makes some aspects of itself available to others. The combination of focus and nimbus results in the mutual level of awareness of objects in the virtual scenario (See Figure 3.26).

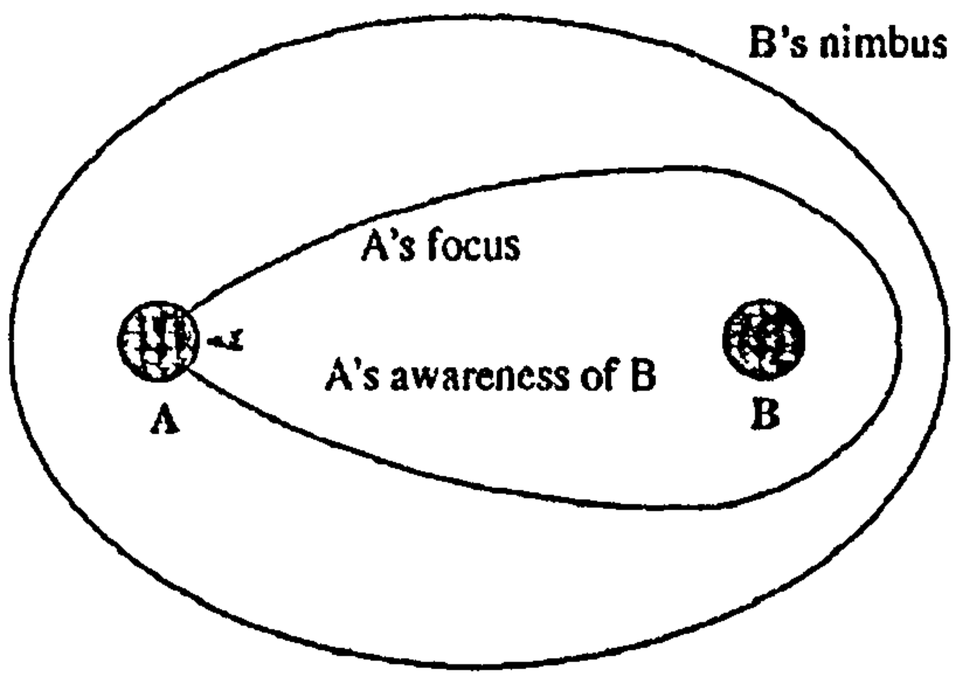


Figure 3.26: Focus and Nimbus define the level of awareness between objects in MASSIVE (Greenhalgh et al., 1995)

Finally *aura* defines the volume of space within which the interaction is possible: two objects start interacting only when their aura collide or overlap (See Figure 3.27). This structured spatial interaction model provides great flexibility to the system and ensures supportability for a large number of users.

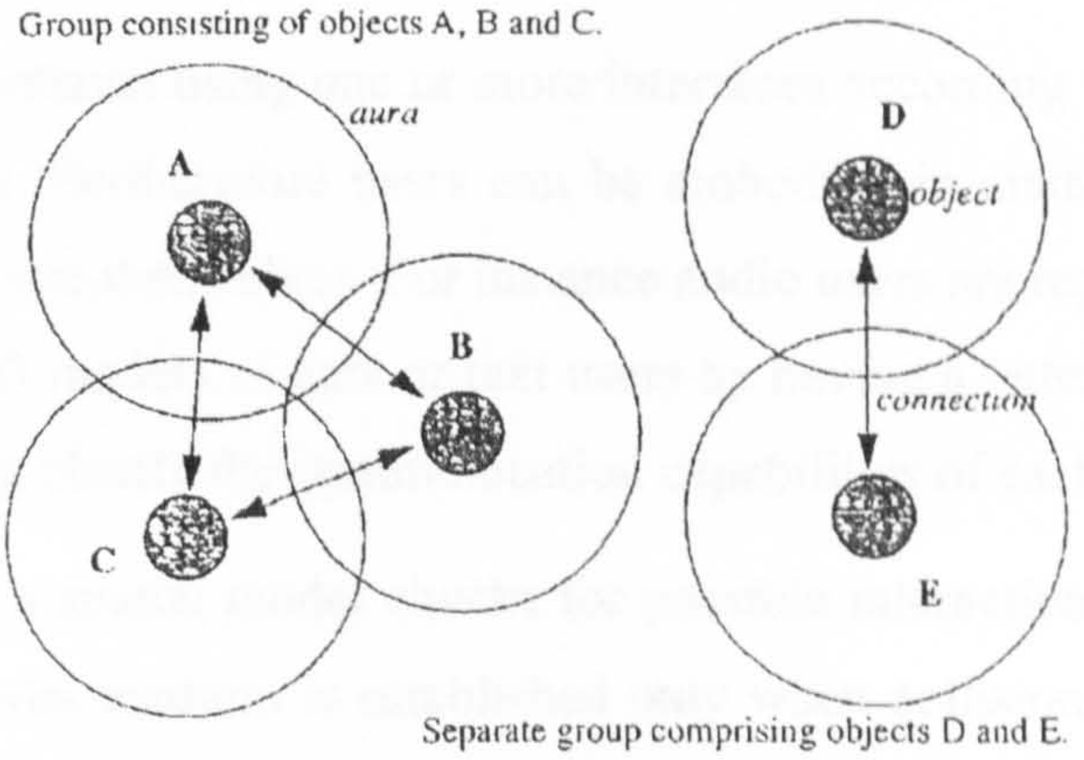


Figure 3.27: Object interactions based on auras in MASSIVE (Greenhalgh et al., 1995)

In terms of functionality, the MASSIVE universe is built upon a set of virtual worlds connected via portals (See Figure 3.28). Users can access one or all of the worlds in the MASSIVE universe freely and they can choose from three different interaction interfaces: *graphics*, *audio* and *text*.

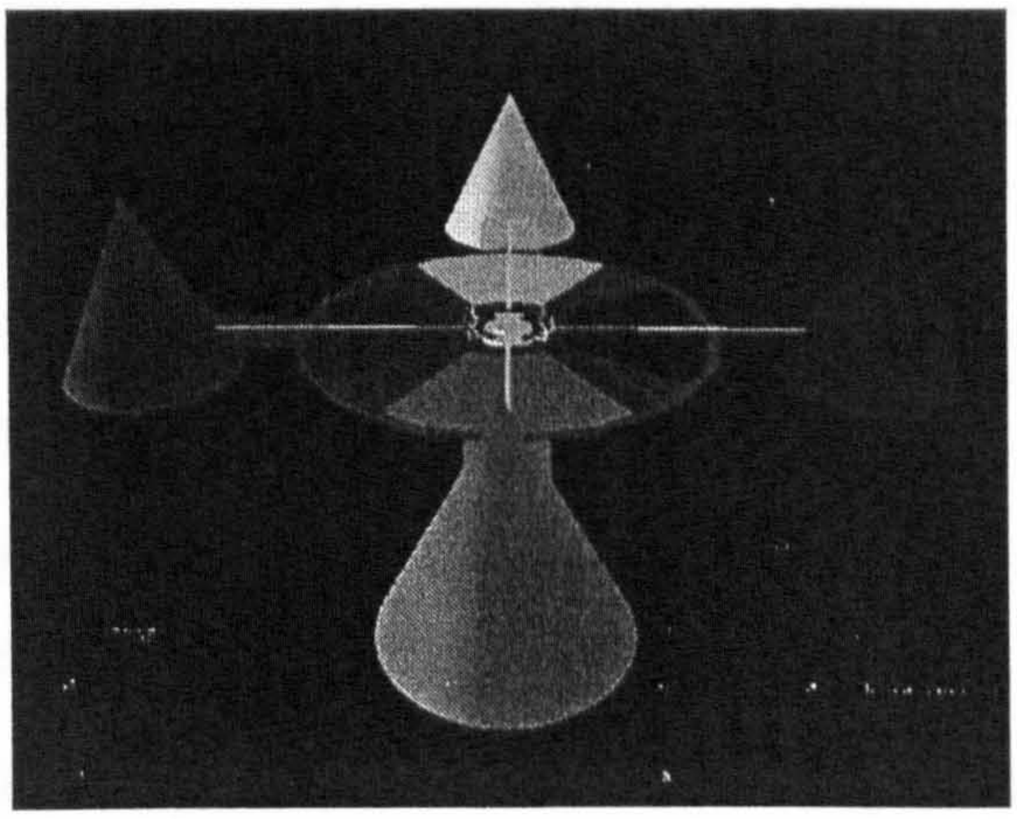


Figure 3.28 : An image of the multi-world structure in MASSIVE (MASSIVE, 2002)

The *graphics* interface renders 3D objects in a 3D virtual space where users can freely navigate. The *audio* interface provides users with both real-time conversation and the playback of stored sounds. Finally the *text* interface allows users to interact through a MUD-like view of the world by sending text messages (See Section 3.4.2).

Users can interact using one or more interfaces according to the performance of their computers. Furthermore users can be embodied in media that they cannot actually display or use themselves. For instance audio users are represented in the 3D virtual scene by 3D models of ears or text users by having a letter "T" close to their heads thus showing clearly the communication capabilities of each participant.

MASSIVE's spatial model checks for possible interactions in each medium. Interaction in a given medium is established only when collision of auras occurs in that medium. Furthermore information transfer, in the three media, between two or more interacting users, is controlled through awareness values computed from focus and nimbus. For example awareness levels in the audio interface are mapped onto volume. Thus volume will be set according to the mutual distance and orientation of the participants.

Aura, focus and nimbus are based on the user's current position and they are upgraded accordingly while the user is navigating the VE. Users can also change general settings for focus and nimbus resulting in narrowing or widening of the area of the objects awareness. Furthermore some physical objects in the virtual scenario can trigger attention and change the spatial interaction settings, for example:

- A podium which extends auras and nimbi allowing the user to address all the users present in the VE
- A conference table which sets aura, focus and nimbus parameters to include the conference table
- A text-to-speech translator which converts text messages into synthesised speech for audio interaction
- A text-to-graphic translator, which displays text messages on a large screen in the graphic medium.

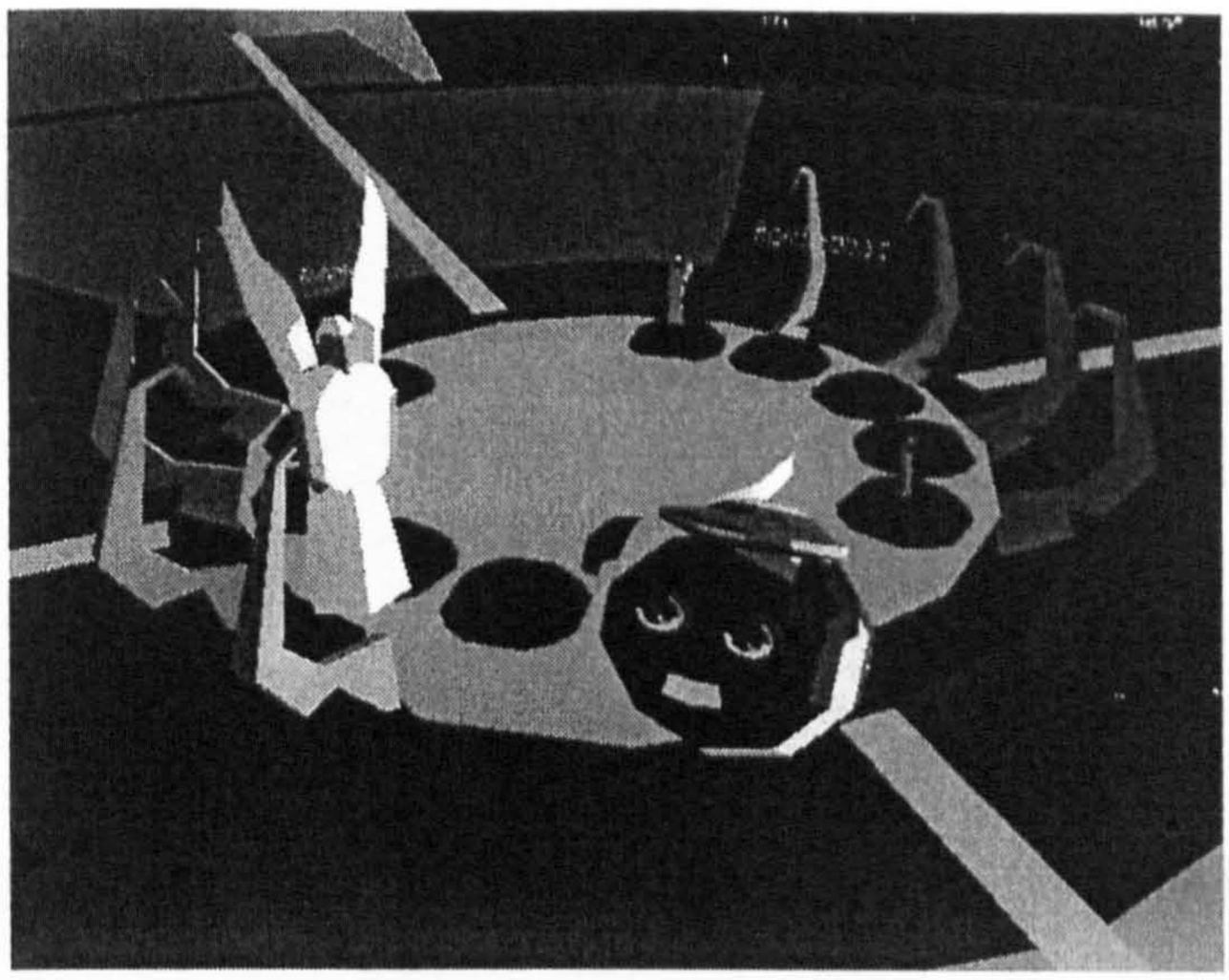


Figure 3.29: A virtual scenario using MASSIVE (MASSIVE, 2002)

Many collaborative virtual environments are based on MASSIVE or later versions of the system and many of these projects were developed at the University of Nottingham where MASSIVE was created (Multimedia Networking for Inhabited Television, 2002; eRENA, 2002; HIVE, 2002).

3.4.4.2 DIVE

The SICS (Swedish Institute of Computer Science) has developed the Distributed Interactive Virtual Environment (DIVE) since 1991. DIVE (Frécon et al., 1998) is a platform for the development of virtual environments, user interfaces and applications based on shared 3D virtual environments.

DIVE especially allows networked multi-user applications, and because of its flexible architecture it has been used as base for many collaborative virtual environment projects (See Section 3.4.3.7).

DIVE allows: the import of several formats such as Autocad, 3D Studio and VRML, the editing of objects such as translation, rotation and scaling manipulation, the handling of texture mapping using a Coloreditor, the assigning of level-of-detail to its objects and it deals with the objects locking mechanisms to ensure data consistency throughout the system.

Figure 3.30 : A collage of images from several projects implemented over the DIVE environment (DIVE, 2002)

Users are represented in the VE through virtual bodies with the capability of movement and posture. Audio and video are also transmitted to the multiple participants.

DIVE focuses on peer-to-peer multicast communication, thus minimising demand for bandwidth especially when continuous streams of data (i.e. audio and video) have to be sent to multiple participants. DIVE also offers several compression methods to optimise the transmission of audio and video balancing quality and network load.

The organisation of the DIVE world is based around a hierarchical distributed database of *entities*. These entities can be compared to objects in object-oriented approaches, although DIVE was entirely written in plain ANSI C.

Whenever something in a DIVE world changes, the modification is first notified to the local database and then the message is distributed to all DIVE users that are viewing this world.

Entity persistence is ensured for as long as a participant interacts with a world. Once the last user of a world has exited, the current world vanishes and its content is stored in a URL file, located in a local database or on remote servers, waiting to be reloaded by the first participant that connects with the world.

DIVE runs on many platforms, including SGI, Solaris, WindowsNT and Linux.

As already mentioned DIVE is a toolkit and many applications and projects were based on it, such as:

- MDraw (Stenius, 1996), one of the first applications implemented on the DIVE system. It is a collaborative virtual whiteboard that allows multiple-users collaborative sketching (See Figure 3.31)
- VR-VIBE (Brown et al., 1996) is an application to browse and search large databases in VR (See Figure 3.32)
- the DIME (DIME, 2002) project, which aims at developing and studying the use of Collaborative Virtual Environments (CVEs) as the base of computer supported meetings for geographical dispersed collaborators
- the DIVEdit (Stenius, 1996) project, which allows the creation of objects and their modification in real time within a multi-user virtual environment (See Figure 3.33).

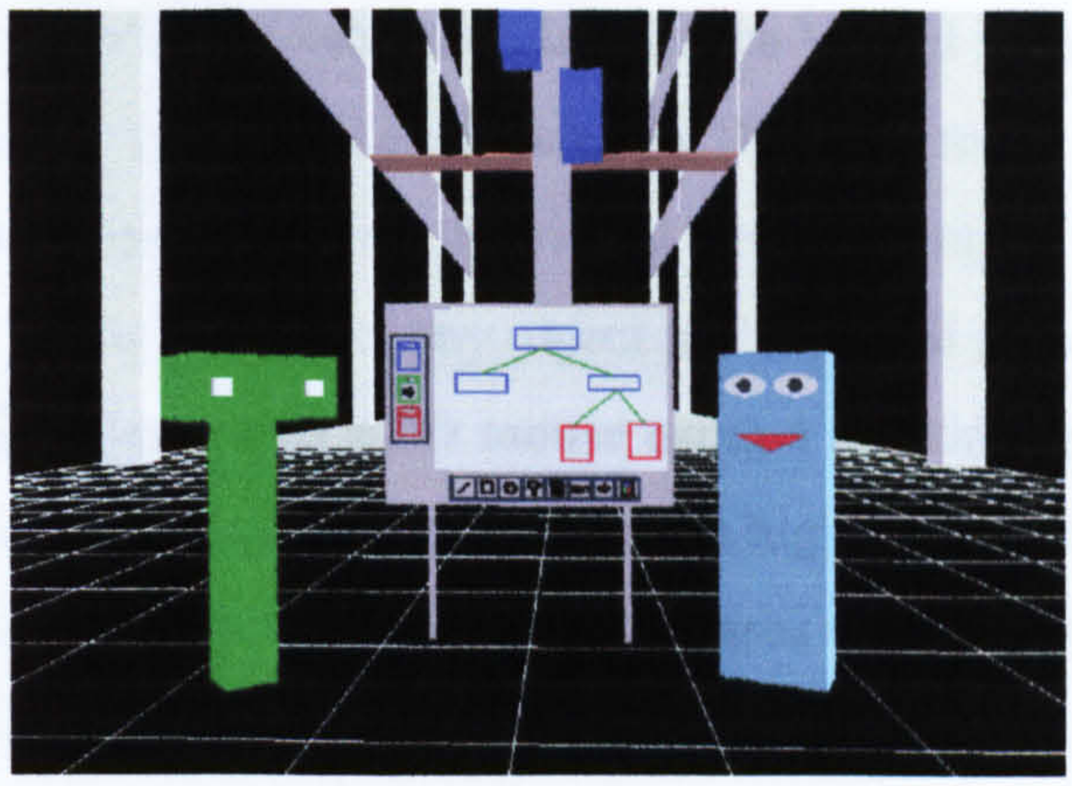


Figure 3.31: A screenshot of MDraw, one of the first applications implemented in the DIVE environment (Brown et al., 1996)

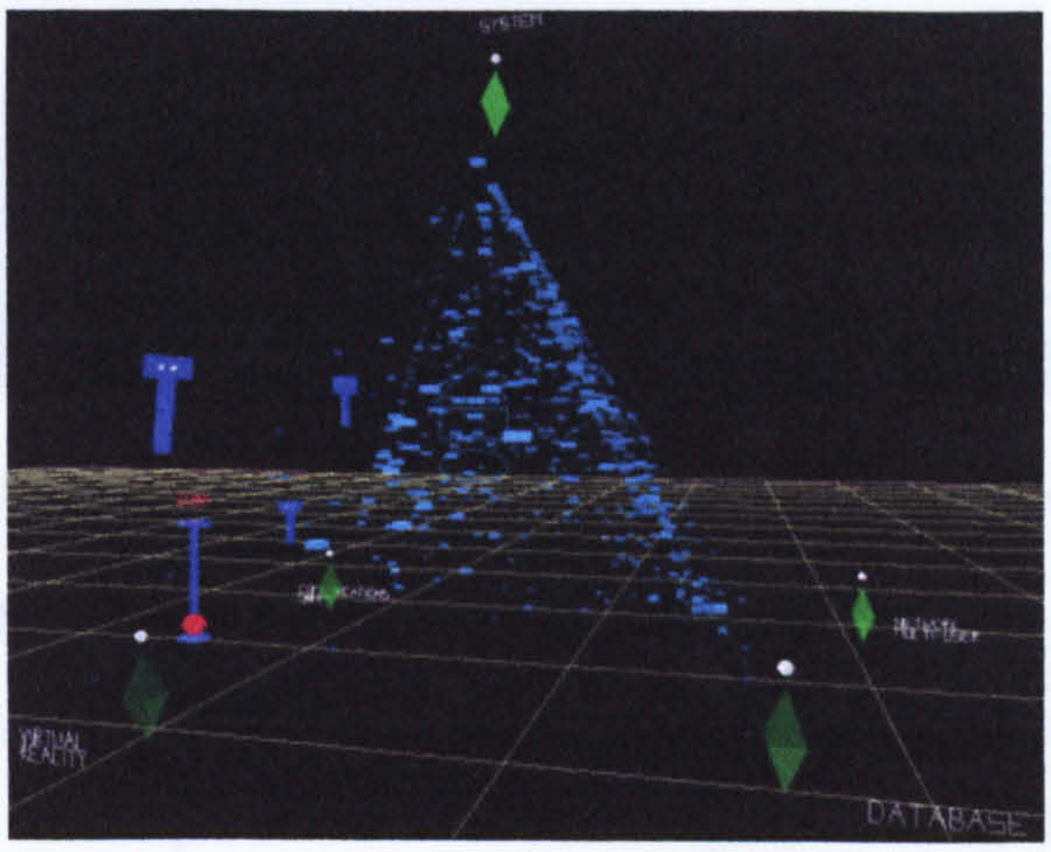


Figure 3.32: A screenshot of VR-VIBE, implemented in the DIVE environment (Brown et al., 1996)

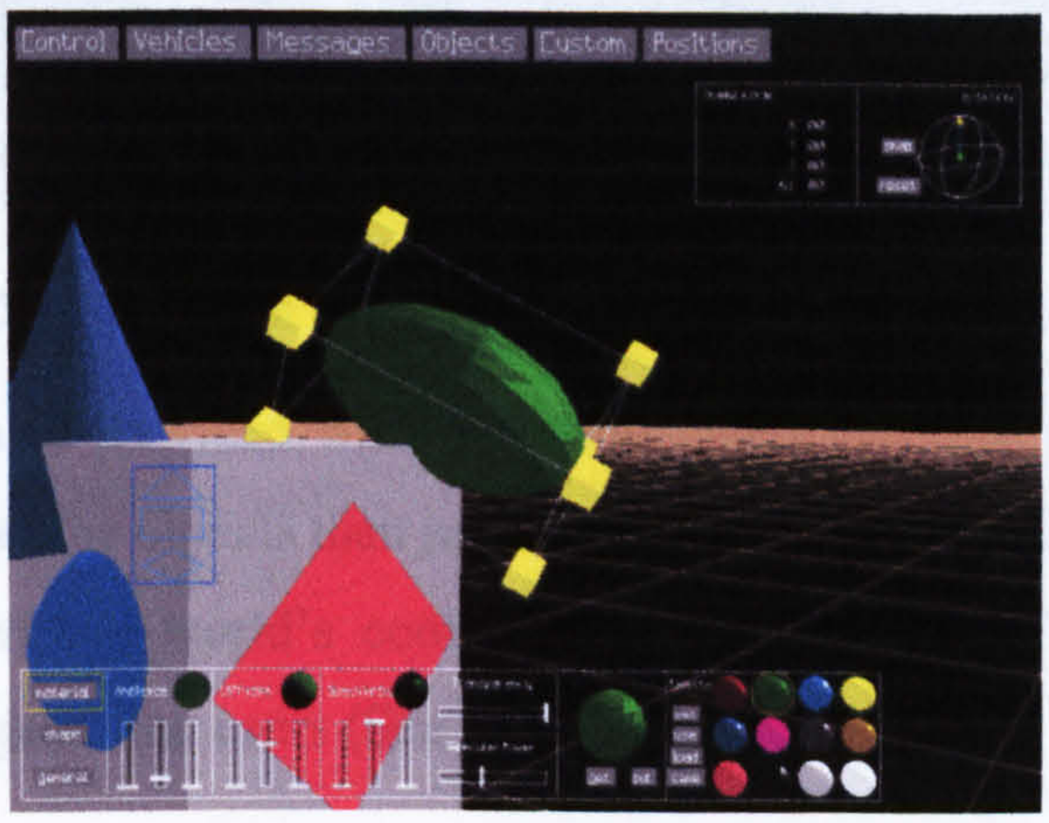


Figure 3.33: A screenshot of DIVEdit, running in the DIVE Interactive Virtual Environment (Stenius, 1996)

In particular DIVEdit (Distributed Interactive Virtual Editor or simply DIVE Editor) (Stenius, 1996) fills any deficiencies in DIVE in terms of the creation of objects in real-time while navigating the virtual environment. DIVEdit allows users to navigate the VE freely and select any object in the virtual scene for modification. The user selects the objects with a 2D mouse using a marking area, which is a box bounded by eight sensitive points at the corners, to highlight the selected object (See Figure 3.33). Objects now bound by the marking area can be rotated, translated and scaled using the "handles".

A material editor handles changes to the surface of the objects. In particular users can change parameters such as ambient reflection, diffuse reflection, specularity and transparency as they wish (See Figure 3.34).

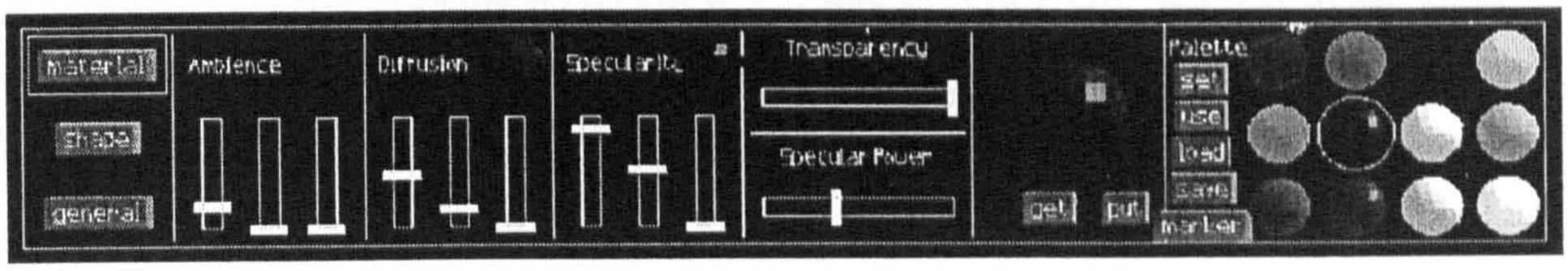


Figure 3.34: The material editor of DIVEdit (Stenius, 1996)

A shape selection panel provides the means for creating new geometries in the virtual scenario. The user can create several basic shapes while he/she is navigating the VE. These primitives are: boxes, spheres, cylinders, cones, lines and ellipses (See Figure 3.35).

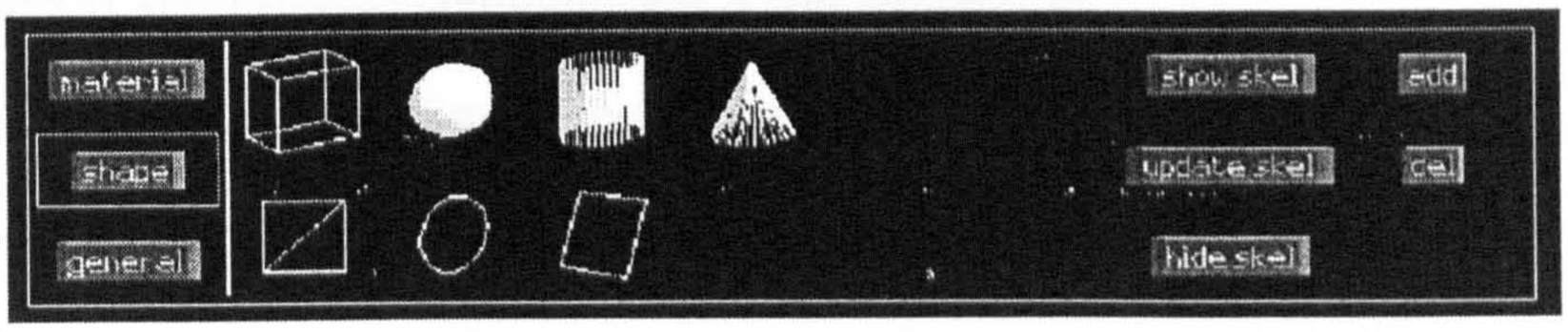


Figure 3.35: The shape selection panel of DIVEdit (Stenius, 1996)

Users are also provided with an object information panel that allows finer position control over the objects than just visual feedback. A virtual compass, always aligned with the virtual world's coordinate system, provides information on the position of both objects and users (See Figure 3.36).

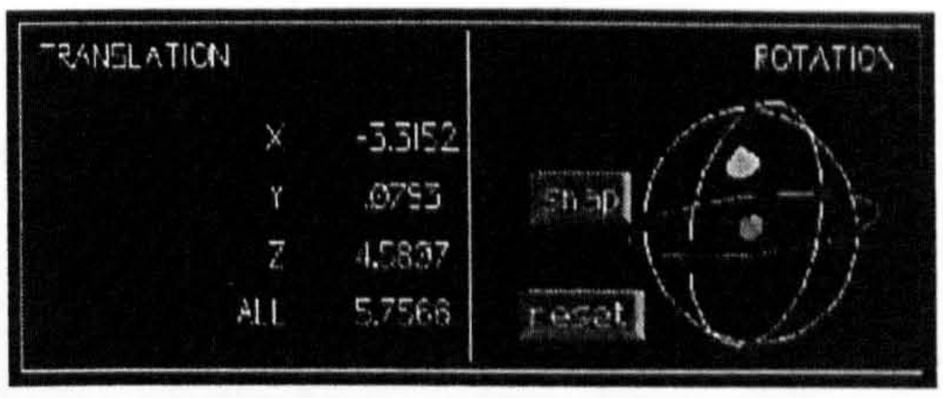


Figure 3.36: The object information panel of DIVEdit (Stenius, 1996)

Due to its real-time creation capabilities DIVEdit could be included in the fifth and last category of CVE systems: Collaborative Virtual Design Environments.

3.4.5 Collaborative Virtual Design Environments

Collaborative virtual design environments (CVDEs) represent the fourth generation of CAD systems (Horváth et al., 2002). They are "the VR application to product and system design activities, allowing the viewing and review of entire systems, assemblies, and parts" (Ragusa et al., 2001, p. 41). CVDE systems enable remote users to share resources, models, and knowledge and allow them to accomplish design tasks concurrently while taking advantage of VR technology to achieve realistic 3D visualization and enhanced interaction.

Horváth et al., (2002, p. 67) found seven basic requirements that form the base of a complete implementation of a CVDE system, which should:

1. Establish a distributed virtual office for remote teams
2. Support synchronous communication and interaction
3. Allow genuine three-dimensional imaging and simulation
4. Support interactive collaboration based on virtual presence
5. Ease creative conceptualisation
6. Allow the sharing of knowledge and put it to work
7. Broker knowledge and expertise for virtual enterprises

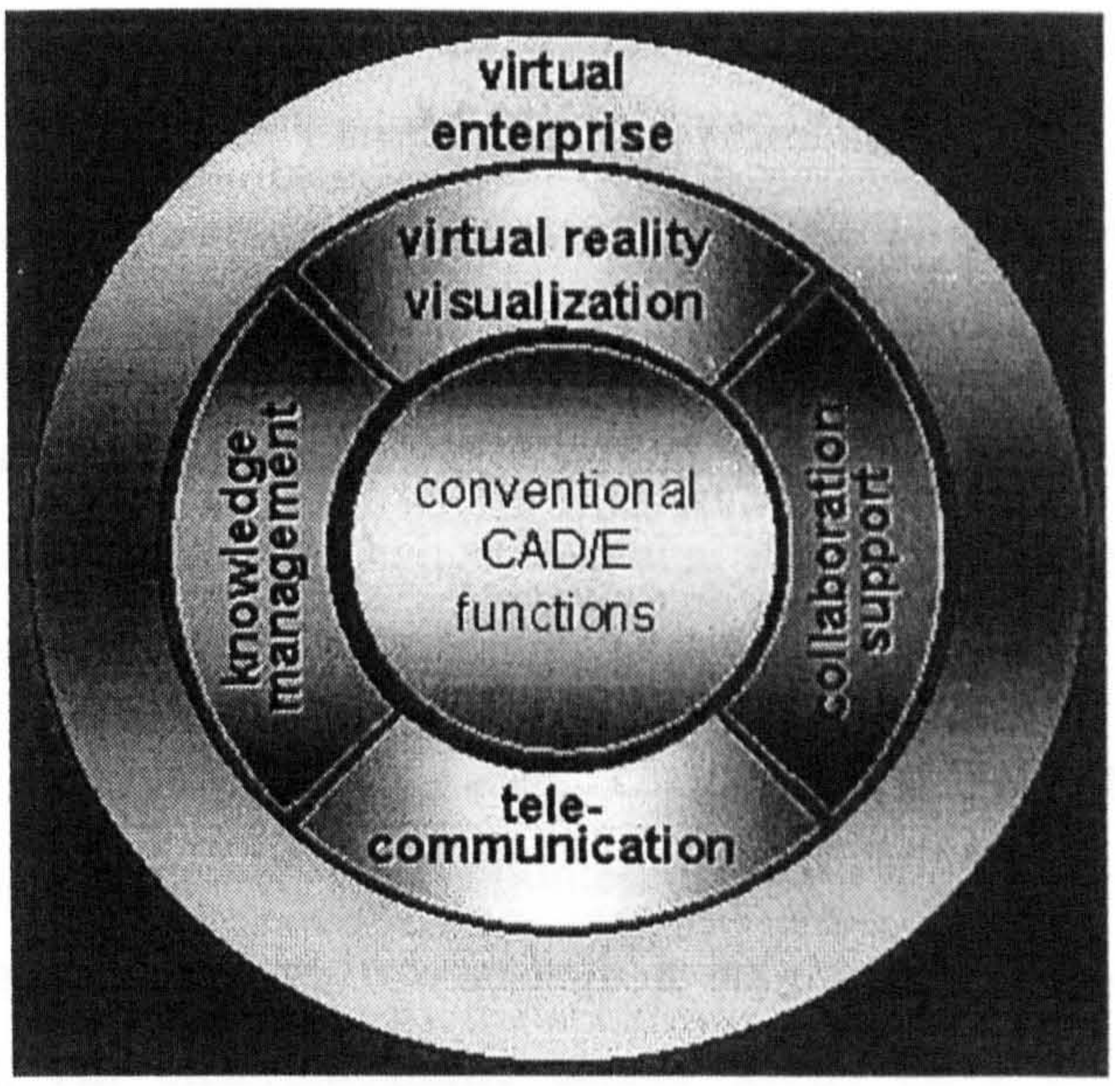


Figure 3.37: Components of a collaborative virtual design environment (Horváth et al., 2002)

Figure 3.37 shows the basic components of a CVDE that enhances the functionality of traditional CAD packages through:

- Advanced interaction and visualization capabilities typical of VR systems
- The use of computer networks as the medium for the distribution of CAD functionality and for connecting the collaborating parties
- The creation of a suite for creative collaboration.

Collaboration support in CVDEs aims to facilitate team understanding and creativity, to support the execution of collaborative functions based on shared models and to bring a virtual presence to remote locations.

The paradigm of CVDE is still in its infancy but many systems can already be included in this category from early experiments to more complex systems. The following sections outline some early examples of CVDE systems together with more recent applications.

3.4.5.1 The Workroom

The *Workroom* is one of the first multi-user VR systems developed for PC's computers and it could be considered an early attempt at a CVDE system. It was

developed in 1992 by Sense8 in collaboration with the Institute for Simulation and Training at the University of Central Florida (Pimentel et al., 1994), and it was unveiled at the Boston Computer Museum in April 1992 (Burdea et al., 1994).

The *Workroom* is an interactive shared virtual environment that allows multiple users to collaboratively create a virtual building using a limited set of pre-constructed elements. Users can change their size in the environment switching from the role of creators to the role of inhabitants of the built environment.

Each user requires two Intel-based PCs and two graphics boards to generate the stereo images in a tracked head-mounted-display (HMD). The interaction with the virtual environment is handled through a six-degree-of-freedom wand and a standard joystick. An electromagnetic sensor keeps track of the position and orientation of the wands to replicate the VE users' actions and interactions with the 3D objects. Communication among the participants is enhanced, through having microphones located inside the HMDs. At the receiving end a 3D sound processor deals with the voice signal and sends it to the headphones located in the HMD as speech.

Users are provided with a virtual wand through which they can interact with virtual objects by creating, editing, deleting 3D elements and assigning materials as texture maps. A locking mechanism assures data consistency and prevents more than one user at a time selecting or editing an object.

The Workroom networking architecture relies on a client-server architecture where the server co-ordinates the simulation activities and upgrades the virtual environment of all the clients and the clients manage the local simulation, handle the mechanical devices such as HMDs and trackers and perform image generation.

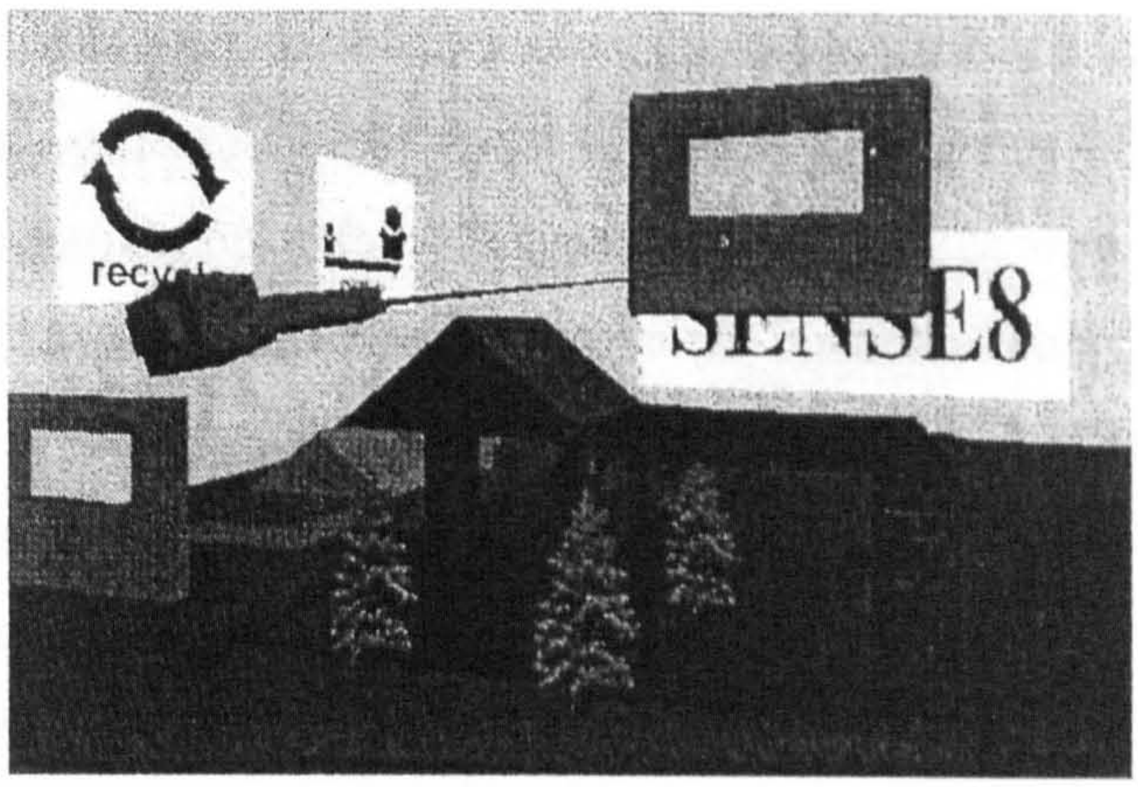


Figure 3.38 : An image of the Workroom VR environment with a user selecting a wall (Pimentel et al., 1994, p. 61)

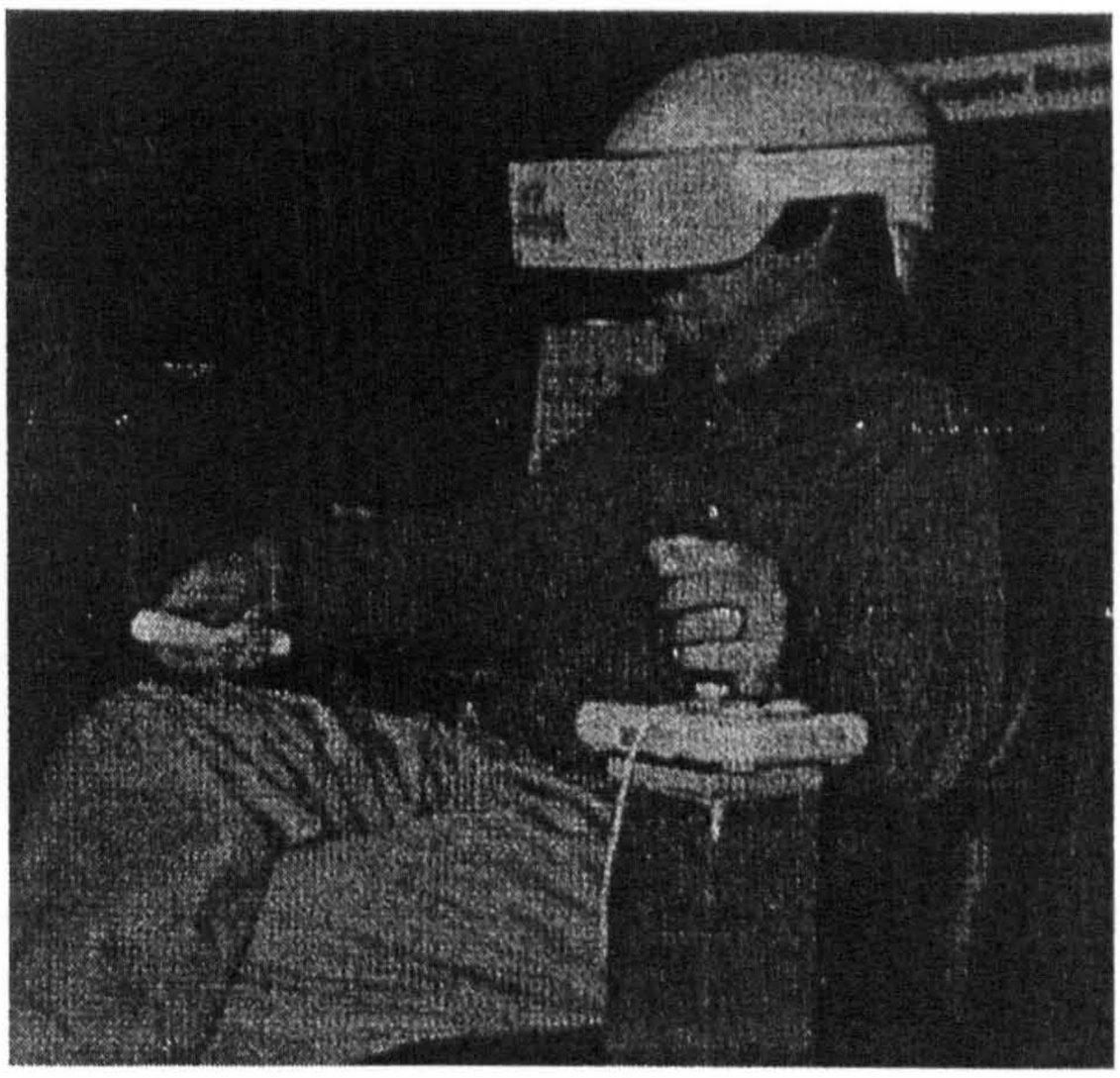


Figure 3.39: The system in use (Pimentel et al., 1994, p. 61)

3.4.5.2 Collaborative Interior Design

In 1994 the European Computer-Industry Research Centre (ECRC) developed a system for collaborative interior design (Whitaker, 1996) that combined the use of augmented reality with the ability to share a 3D model over the network. The scenario of this application was of an office manager working collaboratively with an interior designer to rearrange the layout of an office. The pair could monitor the state of the room from a camera viewpoint and interact with manufacturers by browsing a furniture database through the net to retrieve information and pictures.

Once the piece of furniture was selected it was possible to drag it into the room using a 3D mouse or move it around the room or delete it. In terms of collaboration the system allowed consultation with colleagues at remote sites who were running the same software. Changes made by one user were instantaneously sent to the other participants and a distributed locking mechanism assured data consistency by allowing objects to be modified by only one user at a time. The 3D graphics and the augmented reality components of the system were implemented with *GSP*, a software platform that combines interactive 3D graphics and computer vision technology that aligns 3D models with real-time video. The collaborative part of the application was implemented using *Facile,* a high-level programming language for concurrent and distributed computing developed at the ECRC.

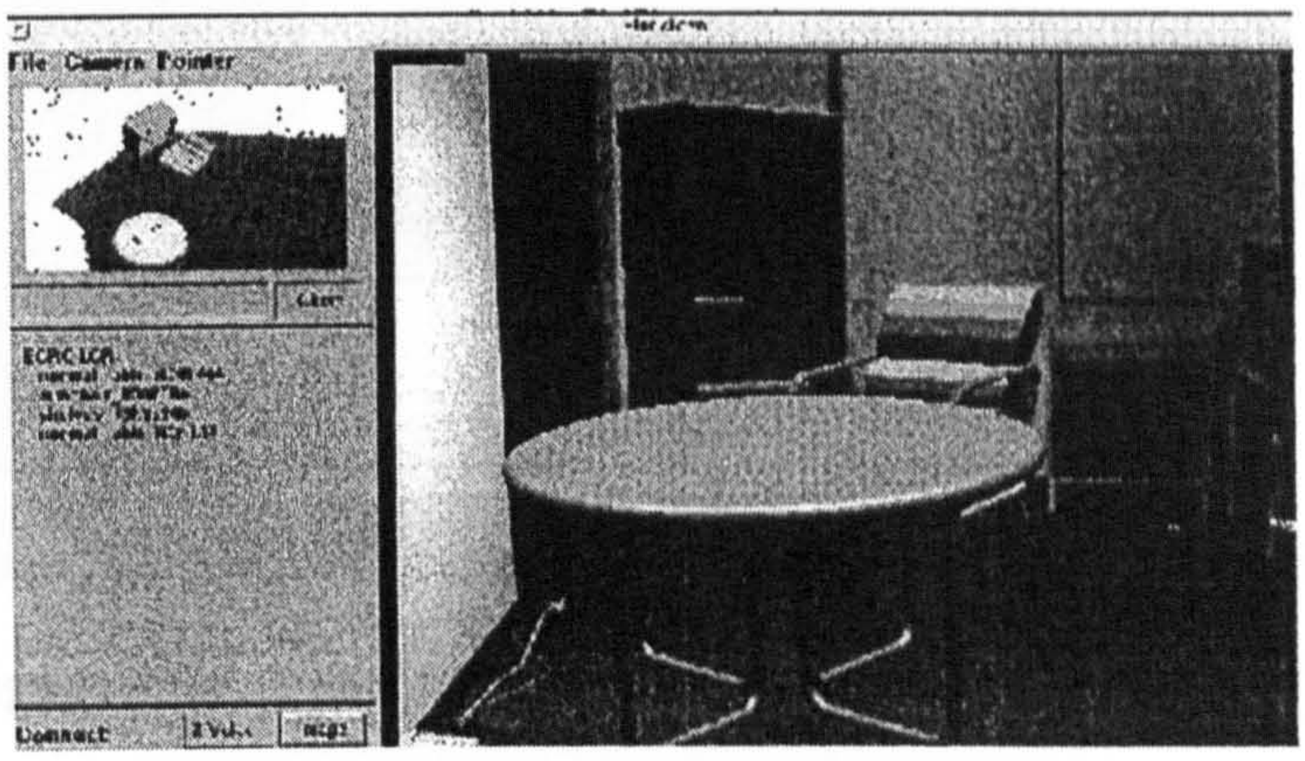


Figure 3.40: A screenshot of the Collaborative interior design application (Whitaker, 1996)

3.4.5.3 A Web-based VRML Collaborative Design Tool for Architects (Yeung, 1997)

In 1997 Yeung (1997) presented a Web-based VRML Collaborative design tool for architects, based on Java™. The aim of the system was to give students and practitioners worldwide the ability to view 3D models and discuss related architectural issues. The system was a web page consisting of two Java™ applets and a display window for 3D models. The client/server architecture allowed retrieval of URL locations of VRML files from the server and it allowed browsing these files

using a database of 3D models. Thanks to the use of Java™ and the web-based approach the entire system was platform independent.

In terms of user-interface the system was divided into three main parts shown in Figure 3.41: the database module on the left; the view port for the visualization of VRML models at the centre-top; and the communication tools module at the centre-bottom.

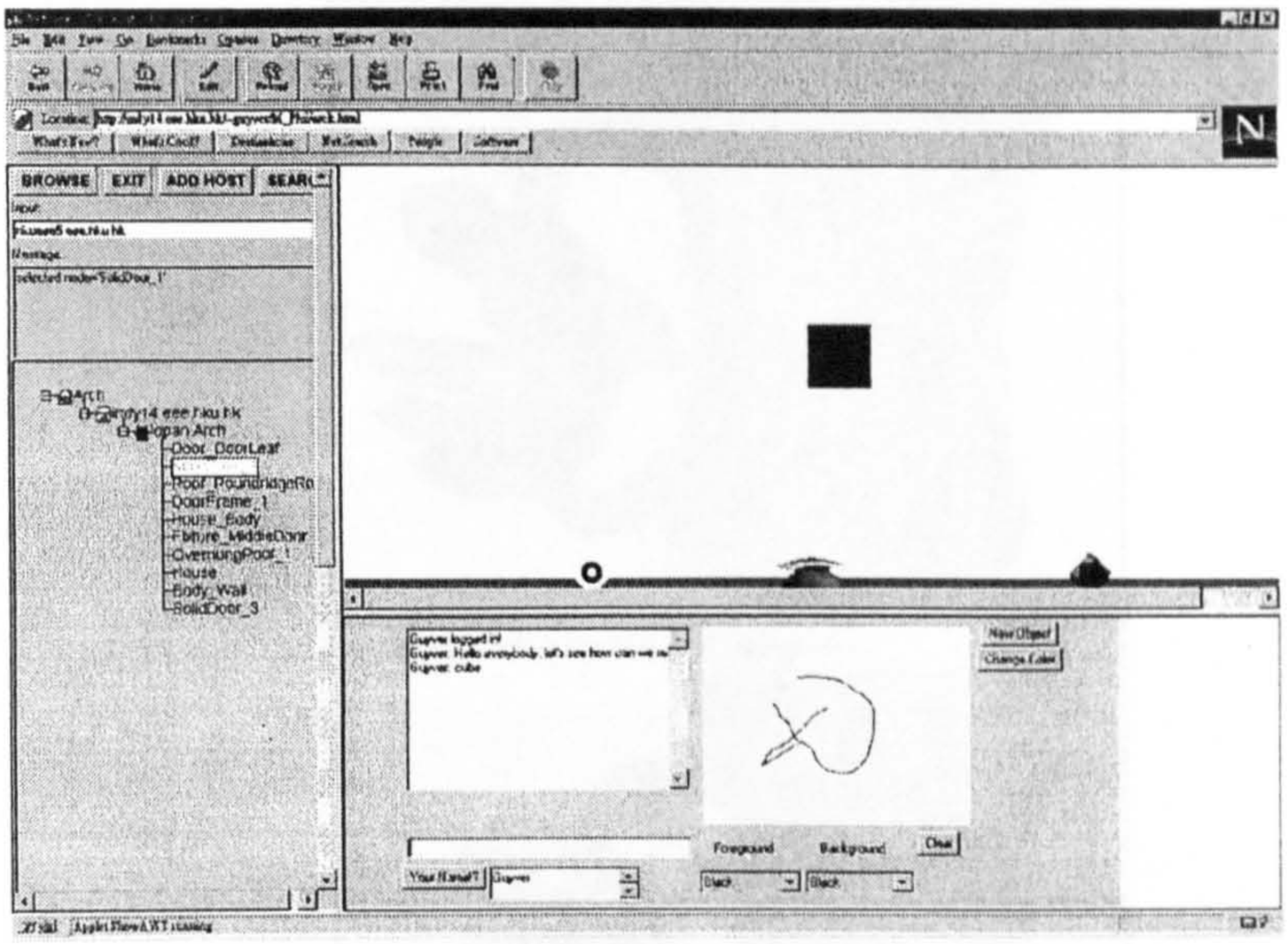


Figure 3.41: A screenshot of the application (Yeung, 1997)

The database module hands the queries to a distributed database of VRML models. The results of the queries were visualized in the VRML view port. Users were also able to make modifications, add new VRML objects and scale and change the colour of the existing models. Synchronous communication with other users was achieved via a chat-line and a virtual whiteboard.

3.4.5.4 VDM

Horváth and Rusák (2001) developed a collaborative system for Vague Discrete Modeling (VDM), a constructive modelling method that represents the shape of the objects created through particle clouds. A particle cloud is an arrangement of particles in a finite region of space within which the 3D model is defined and therefore the 3D model is called "vague". The advantage of this approach is that designers no longer have to assign precise geometrical parameters to

free form or not precisely defined shapes. The participants are all represented in the virtual space by models of hands, each having a different colour, and they can synchronously create vague shapes via gestural sweeping. The system also supports interactive modification of the vague models, in ways such as sculpting, through which designers can collaboratively add or remove virtual parts from the vague model.

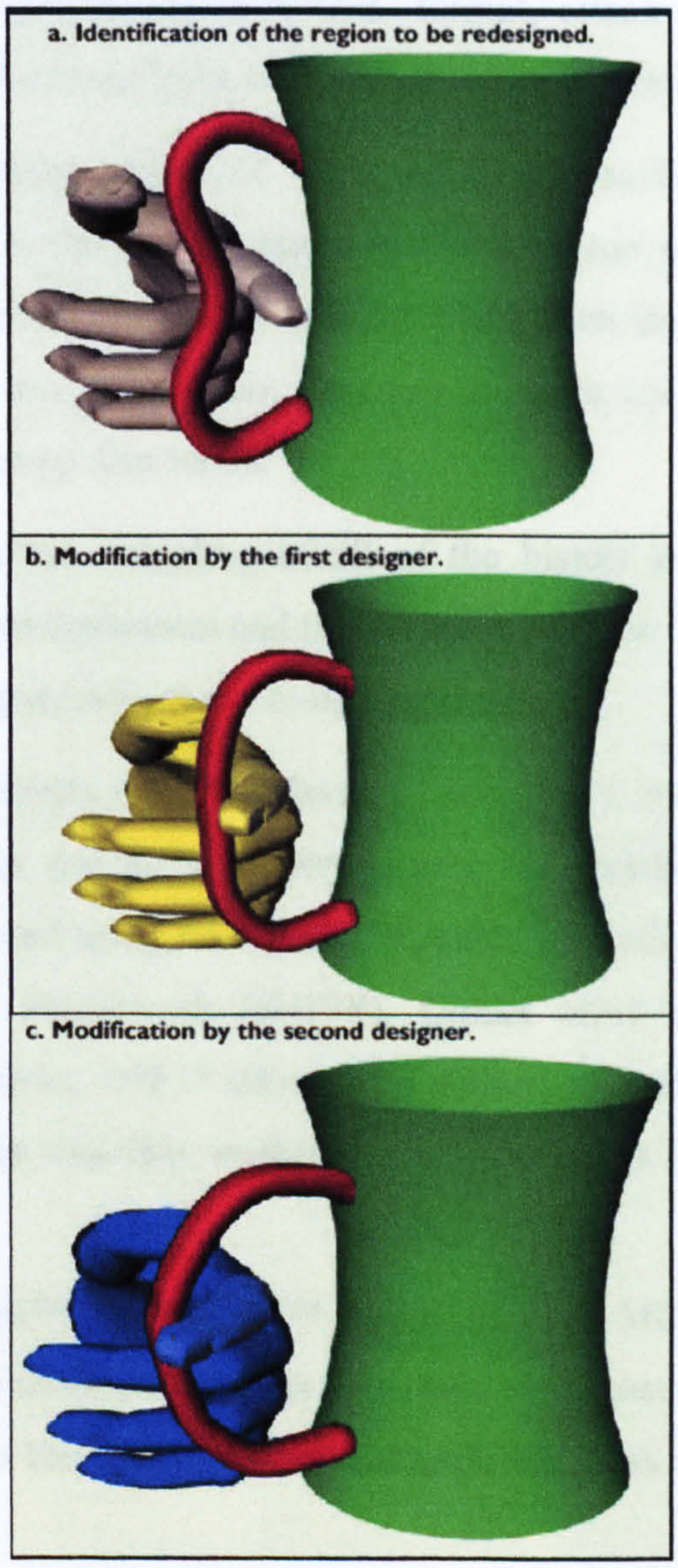


Figure 3.42: Multi-user interaction in VDM (Horváth et al., 2001)

3.4.5.5 MUG

The MUG (Multi-User Groups for Conceptual Understanding and Prototyping) system (MUG, 2002 and Foster et al., 2001) is a collaborative 3D

environment to support knowledge-based conceptual design that lets users author the structural, behavioural and functional knowledge of a design in a 3D virtual environment.

This application was developed to help at the conceptual design stage while working in a shared environment. Users can not only define geometries, such as blocks, tori, spheres, cylinders and NURBS-based curves and surfaces, but they can also import pre-existing geometries in VRML format, attach functions to these objects and then define functional links with the other components of the design.

From a collaborative point of view MUG is based on client/server architecture and thanks to this its synchronous collaboration is assured over the network. During the collaborative session, all the changes on the objects caused by the users are sent to the other active users. Communication is assured through an IP-based audio channel for group discussion.

MUG also allows the recording of all of the history of the collaborative session, including the audio discussion and the design generation. This feature can be used as a reference to help document the design process.

The system was entirely written in Java™. Specifically Java3D™ was used to build the 3D environment and the geometry kernel, the collaborative side of the application was implemented using JavaSpaces™, while the audio conferencing tool relies on Java™ Media Framework (JMF™). Unlike other voice conferencing packages that are peer-to-peer, JMF™ allowed the authors to implement a "many-to-many" audio conferencing tool that worked perfectly on both Suns and Windows operating systems.

The choice of Java™ and Java3D™ as the basis for MUG, instead of C++ and OpenGL, which were usually used, was motivated by its ease of integration with other supporting packages for shared worlds and multimedia, as well as by its cross platform abilities.

MUG was used as a test application to prove the concept that a sophisticated, multi-platform collaborative modelling environment, also capable of interactively handling complex CAD models could be created entirely in Java™.

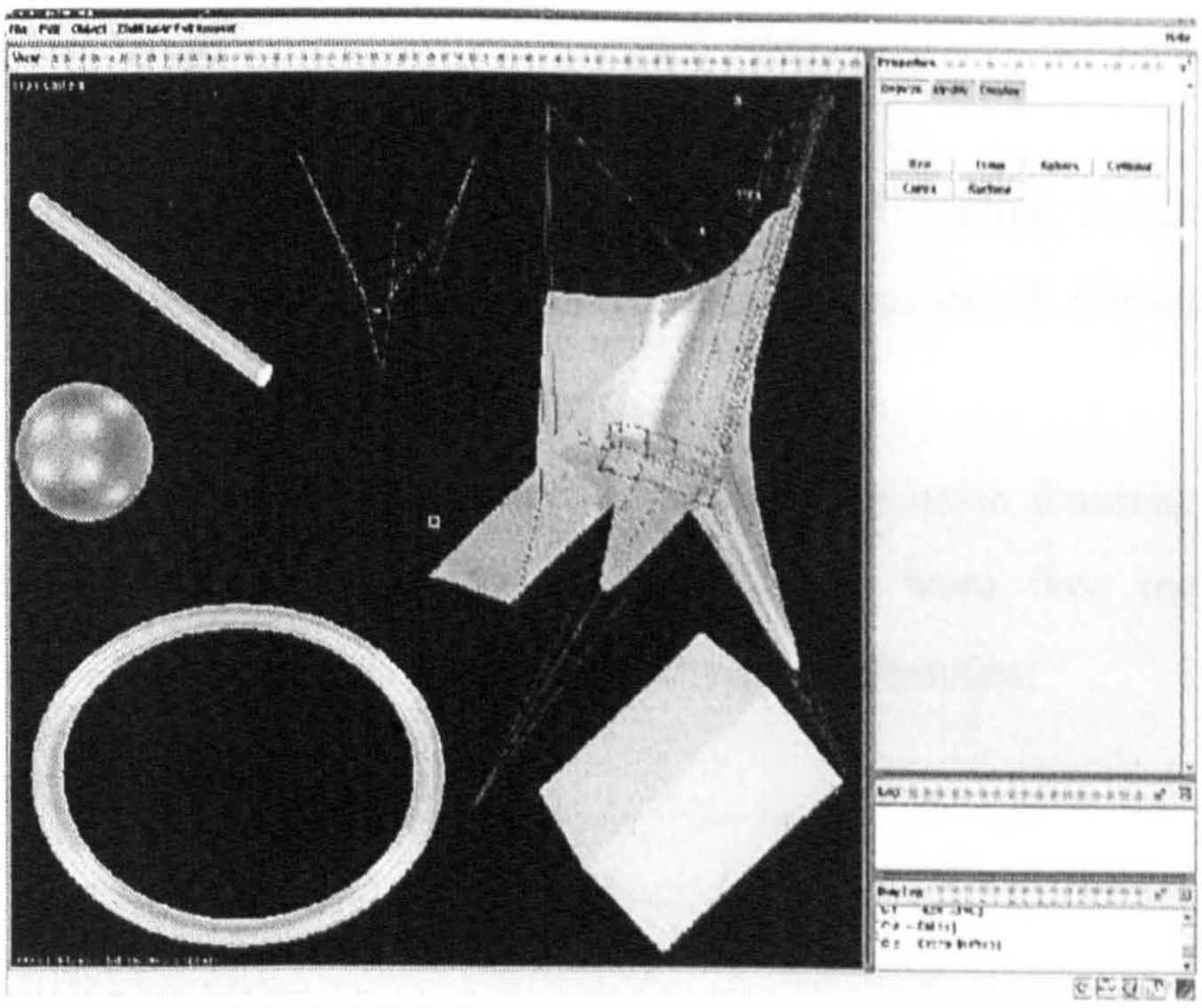


Figure 3.43: A screenshot of the MUG interface (Cera et al., 2001)

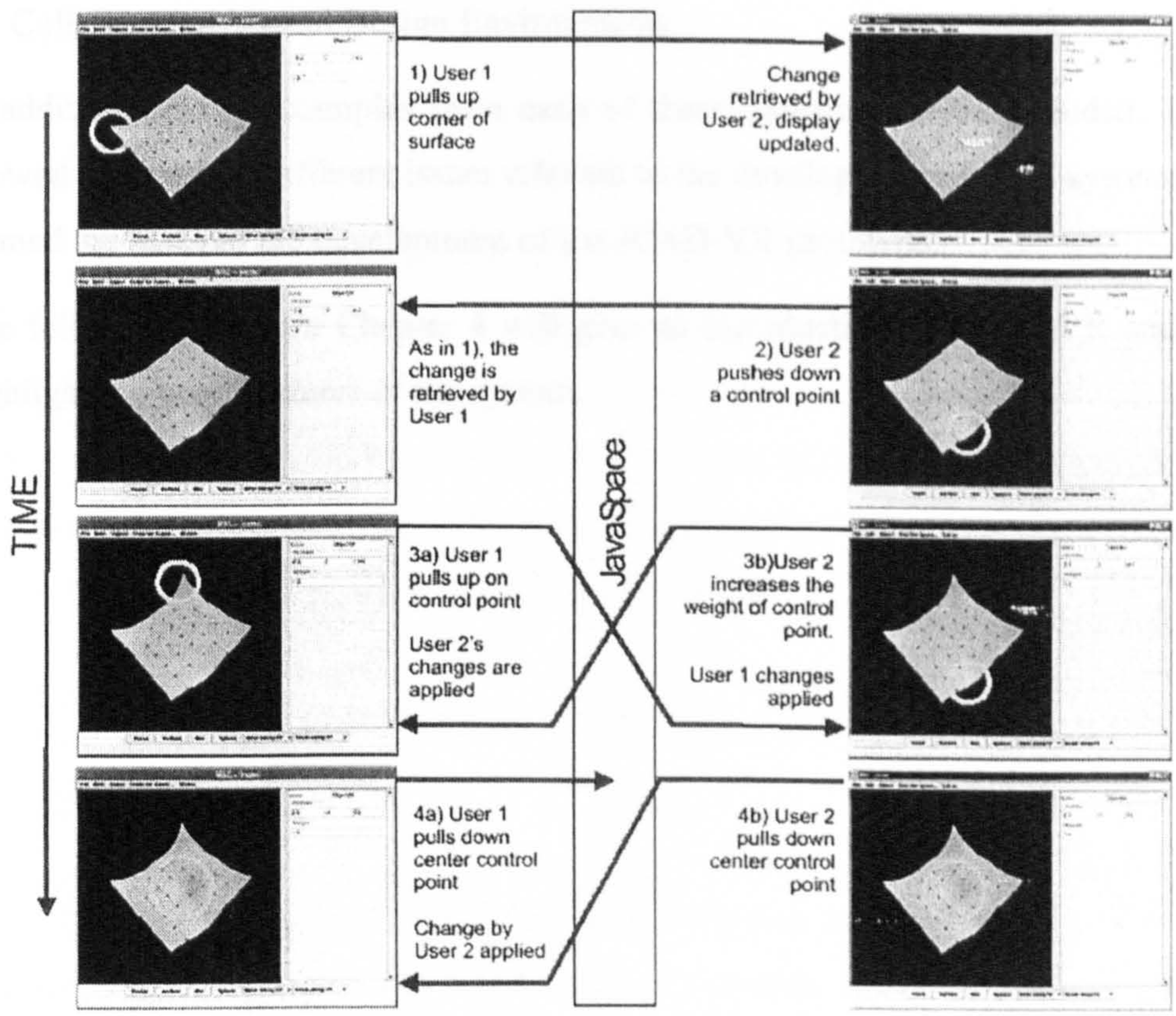


Figure 3.44: A collaborative sequence from MUG (Cera et al., 2001)

3.5 Conclusions

This chapter gave an overview of the use of Virtual Reality in creating collaborative environments. CVEs extend the capabilities of VE allowing concurrent collaboration by remote participants.

The first sections of the chapter highlighted the basic features of CVEs and then suggested their categorisation by type. There were five main categories identified, depending on the systems' architecture and features:

- Large distributed virtual simulators
- Text based shared VE: MUDs
- Multiple user navigation systems
- Platforms for implementing VR shared applications
- Collaborative Virtual Design Environments.

In addition, several examples from each of these categories were included. These showed some of the different issues relevant to the development of CVE systems and formed the basis of the development of the JCAD-VR prototype.

The following common Chapter 4 will give an introduction to JCAD-VR and will highlight the main features of the system.

4 Development of a Framework: JCAD-VR (Java™ Collaborative Architectural Design Tool in Virtual Reality)

4.1 Introduction

The previous chapters have provided an overview of current research in the field and have supplied the theoretical background to the following chapters. This chapter will introduce the proposed research framework and show the development of the working prototype called JCAD-VR: Java™ Collaborative Architectural Design tool in Virtual Reality.

The aim of this chapter is to show the overall structure of the research framework and to provide a broad picture of the entire system. Technical details along with theoretical and practical choices will then be discussed in depth in the following two chapters.

The first part of this chapter provides reasoned justification for the way the framework was developed, it introduces the system and it outlines the technical choices. The second part provides a general introduction to the framework, to its features and to its architecture as well as a description of the number of modules that comprise JCAD-VR. The end of the chapter summarises the scope of the research and gives consideration to the technical choices arising from the project.

4.2 VR as a Collaborative Design Tool for Conceptual Design

Conceptual design at the initial creative phase of the design process is a complex activity characterised by the use of intuition rather than mathematical formulae. During this stage designers try to give concrete form to their abstracted models through mental simulation. Designers traditionally make sketches as a way of doing this, exploiting their abstract and ambiguous nature to explore new design

solutions. As Shukur (2000) notes, sketches on paper are a natural and intimate medium that allows the down/uploading of complex information to and from the designer's brain, they are also easy to correct or change, portable, extremely flexible and they can store information conveniently and keep track of ideas and solutions.

During this complex undefined phase computer-based systems are not usually employed due to the discrepancy between the designers' conceptual models and the limits of present-day Computer Aided Design (CAD) interfaces. These bind the designers' freedom through their imposition of a formal and constrained graphic language rather than them supporting uncertainty, flexibility and dynamic manipulation. Consequently, as Turner et al. (1999) note, this means designers have a large conceptual gap to bridge between their abstract and possibly vague mental representations in sketches and the formally defined shapes created by current CAD systems.

In addition, the CAD industry often trades the qualities of having advanced geometrical control for the qualities of interactivity and usability, therefore delivering complex interfaces that often lack in flexibility and user-friendliness. The effect of this is that although CAD systems can feature a great number of functions allowing the modelling of complicate shapes, they cannot be used at the very early stage of product design where pencil and paper are still the most effective. Commercial CAD systems are designed to fulfil the needs of a wide range of engineering fields and to support the construction process, they have not been designed to promote the creative acts typical of the initial stages of design.

As a consequence, the early design phase is disassociated from the product definition stage that uses CAD/CAAD because the designers are forced to adopt traditional tools like paper and pencils. This creates an amount of inefficiency within the design cycle and consequently higher production costs. There is an evident need for innovative computer-based tools specifically tailored to support creativity within the design community.

In the last few years, the growing awareness of more natural, user centred form of interaction, has promoted interest in expanding the field of Human-Computer Interface (HCI). As a result, a new generation of computer-based

applications has embraced innovative technologies whose adoption in turn, has fostered original interfaces aimed at promoting user-friendliness and more interactive behaviour.

The speed at which technology is evolving is bringing the application of VR technologies within the sphere of the design professions. The increasing growth of computational resources and hardware power is facilitating a transition to the use of desktop VR applications as truly feasible tools for everyday use.

As a consequence, some design and manufacturing companies have already started to investigate how VR could be used effectively within the design process. Although VR is now a mature technology, it is seldom used in architecture during the design process it is more often merely used as a powerful presentation technique. Virtual worlds are often created using CAAD/CAD packages and deriving world representations by conversion, in this way, becomes a time consuming and therefore expensive task.

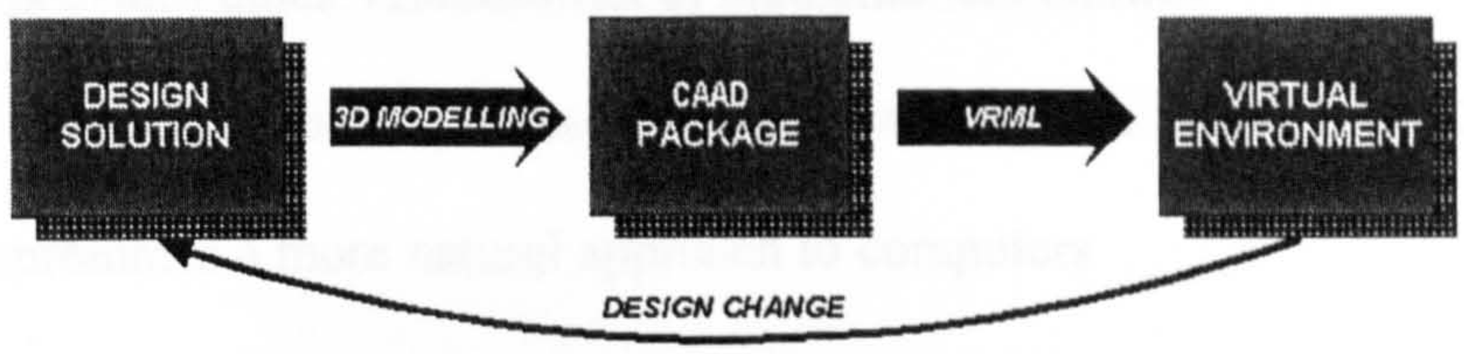


Figure 4.1: The traditional scenario showing the relationship between the 3D-modelling phase and VR

A virtual model, if it is employed, is usually only created at the end of the design stage when most key decisions have been evaluated, and mainly to impress contractors and clients (See Figure 4.1). It is therefore evident that under these circumstances the use of VR is inefficient and consequently leads to growing costs rather than fostering creativity and ultimately improving design quality.

With the benefits brought by its highly dynamic nature, the early use of VR could allow the designer to study design solutions directly through the manipulation of simple shapes in a virtual space, and hence successfully combine the creative and modelling stages (See Figure 4.2).

Figure 4.2: The proposed scenario

It is also highly predictable that in the near future VR will become the interface for the next generation of Computer Aided Drawing (CAD) applications described in research literature as Virtual Reality Aided Design (VRAD) systems. VR-based systems have the potential to become highly effective tools letting the user explore design solutions in a more intuitive and natural way. The use of VRAD tools has a number of potential advantages over traditional CAAD systems:

- It provides real-time interactivity
- It involves the designer's imagination to a greater degree
- It provides quick visualisation of mathematical entities
- It enhances creativity in the conceptualisation phase (Sener et al., 2002)
- It promotes a more natural approach to computers
- It helps designers make the transition to digital format earlier
- It has the potential to shorten the time taken to manufacture physical models through rapid prototyping processes (Ucelli et al., 2000)
- Ultimately it improves design quality by increasing the level of control on the design of the product.

Another crucial problem of current CAAD systems is their complete absence of any form of synchronous collaboration between architects within design teams. With the present CAAD technology there is little room for multi-user interaction during the creation of a 3D-model since most systems allow only asynchronous collaboration. The recent growth of network-based virtual communities however has brought a new level of complexity to the notion of virtual spaces by providing the technology for remote presence and collaborative experiences.

The research described in the ensuing chapters takes advantage of these new emerging technologies through the development of a system specifically designed to help architects in the initial stages of the design process. The JCAD-VR framework pursues the articulated vision of VR as an instrument developed to assist the participants in the design process – the professional or client body - during these early stages. It is a collaborative design tool developed to ease the initial stage of the design process providing the capability to create 3D-shapes and to share them among the users in the virtual world.

4.3 Justification of the Project

As noted by Lawson (1990), design methodologists in the past agreed on the need for iterative cycles between several phases of the design process. From studies of designers' behaviour many authors observed that there were an indefinite number of return loops from the moments when the gathering of information and structuring of the design problem took place, known as *analysis*, to the time when design solutions were generated, known as *synthesis* (See Figure 4.3).

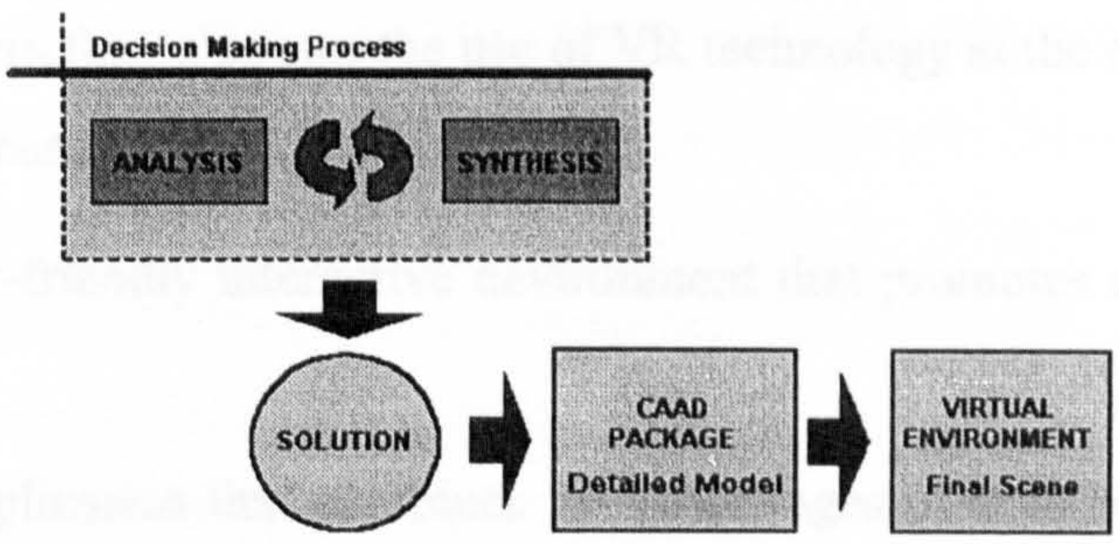


Figure 4.3: The use of CAAD and VR in the traditional decision making process

The development of JCAD-VR provides the designer with an appropriate, quick and practical response to their need for iteration in their search for design solutions. Therefore JCAD-VR provides the means for a more effective use of VR at the very beginning of the decision making process thus helping architects bridge the gap between the *analysis* and *synthesis* phases (See Figure 4.4).

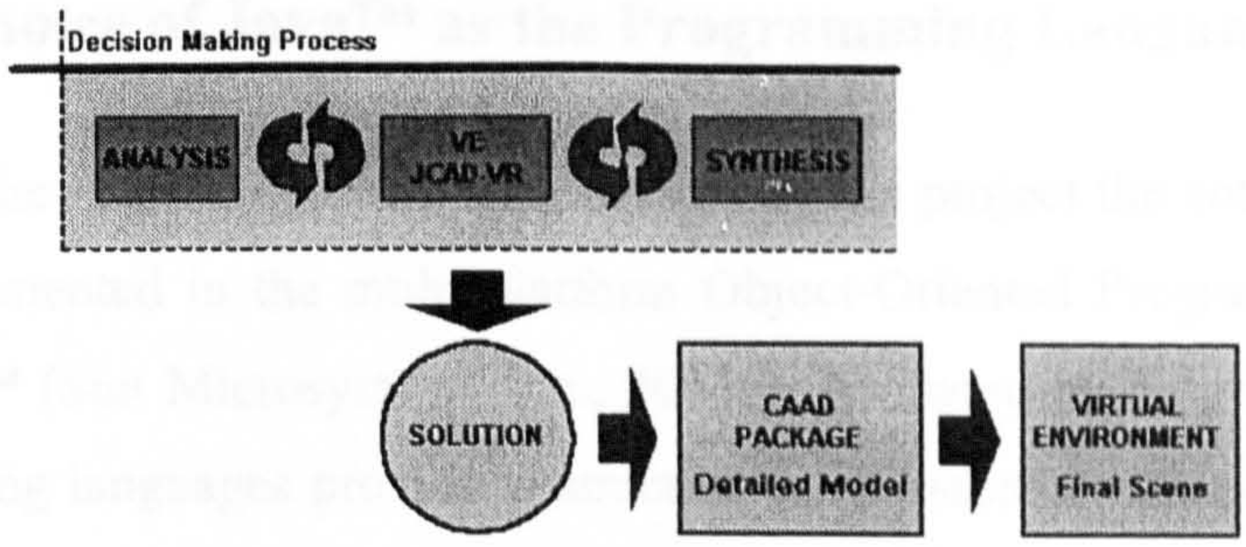


Figure 4.4: The role of JCAD-VR in the decision making process

The development of the JCAD-VR framework proposes an innovative approach to deal with some important issues:

- The use of VR in the early stages of the design process
- The traditional complexity of creating a VR environment
- The lack of a VRAD tool specifically tailored for architects' needs
- Lack of true, synchronous collaboration in commercial CAAD packages.

JCAD-VR was designed to tackle these issues through a number of characteristics and features:

- It is a system that proposes the use of VR technology at the early stages of the design process
- It is a user-friendly interactive environment that promotes a visual approach to design
- It is an application that combines the advantages of both VR and Computer Supported Cooperative Work (CSCW) systems.
- It supports communication through multimedia features
- It has an overall architecture that encourage synchronous collaboration
- It is intrinsically cross platform and hardware independent promoting platform portability.

4.3.1 The Choice of Java™ as the Programming Language

Due to the amount of flexibility required by the project the entire framework has been implemented in the multi-platform Object-Oriented Programming (OOP) language Java™ (Sun Microsystems Inc., 2002a). As stated by Eckel (2000, p. 30) "all programming languages provide abstraction [...] Assembly language is a small abstraction of the underlying machine. Many so-called 'imperative' languages that followed (such as Fortran, BASIC, and C) were abstractions of assembly language".

Object-Oriented Programming (OOP) languages, such as Java™ or C++, have introduced a new level of generalization. In contrast with traditional procedural languages where the level of abstraction provided still forces the programmer "to think in terms of the structure of the computer rather than the structure of the problem" (Eckel, 2000, p. 30) OOP languages help the user develop code in a manner that is closer to the mental process that the programmer would normally follow.

As Eckel (2000, p. 31) notes, "the idea is that the program is allowed to adapt itself to the lingo of the problem by adding new types of objects, so when you read the code describing the solution, you're reading words that also express the problem. This is a more flexible and powerful language abstraction than what we've had before. Thus, OOP allows you to describe the problem in terms of the problem, rather than in terms of the computer where the solution will run". These *objects* are self-enclosing elements of the software that can be used to accomplish a certain task and whose mutually logical relationships with other objects significantly facilitates code development and efficiency through code reuse.

More specifically, Java™ is a last generation mature and complete Object-Oriented programming language that is extremely flexible and relatively easy to implement, deploy and maintain. It is intrinsically multi-platform and it supports multithreading and multiprocessing making it possible to develop programs that run on simple PCs as well as supercomputers (Sun Microsystems Inc., 2002a).

A number of features made Java™ the perfect choice for the development of a VR-based multimedia collaborative environment such as JCAD-VR, these include its:

a) Multi-platform nature

b) Network-oriented architecture

c) Easy database management

d) Availability of existing APIs dealing with:

- 3D Graphics through Java 3D™ (Sun Microsystems, 2002b)
- Audio and video support and real-time network streaming through Java™ Media Framework (JMF™) (Sun Microsystems, 2002d)
- Networking within standard Java™ 2 Software Development Kit (Sun Microsystems, 2002c).

Although it is sometimes less efficient in terms of performance if compared with other languages, the adoption of Java™ as the programming language offered great flexibility, true scalability and last but not least, complete multi-platform support. The system demonstrated Java™ platform's independence successfully when tested in different configurations where PCs and Sgi workstations were concurrently used. The network-centric nature of Java™, and its multimedia integration and multi-processor support make it the obvious choice for the development of a real-time multimedia collaborative system.

4.4 JCAD-VR: The Framework

As described the overall JCAD-VR framework has been developed to allow dynamic interaction with the virtual environment during the early stages of the design process. The following paragraphs will provide an overview of the functions and of the architecture of both the overall framework and the working prototype. The description in this chapter provides a general picture of the system while specific details of the technical implementation will be provided in the following two chapters.

4.4.1 The Idea behind it

JCAD-VR has been developed based on two main ideas:

1. All the users present in the virtual world have to be able to share the very same virtual environment in a "transparent fashion"
2. Instead of being based on traditional WIMP (Window, Icon, Menu, Pointer), the Human-Computer Interface is part of the virtual world itself where each element of the HCI becomes an object belonging to the 3D world and so can be manipulated like any other entity within the environment: the HCI becomes a *3D-HCI* perceived as part of the virtual world itself.

The first point results in the development of a collaborative architecture for the system. Every user that logs in can seamlessly interact with the environment, communicate with other users and manipulate objects while the system ensures that the environment in the background is fully consistent across all the users.

The second point has totally influenced the way the Human-Computer Interface has been developed. The implications of an interface that is a part of the virtual world are twofold:

1. From the technical point of view, once the interface has been designed, it becomes independent from the visualisation device used. The system can therefore be easily adapted for different devices by just rewriting the code that is handling the device. No matter whether the application is running on a simple screen, on a Reality Center™ or linked to an HMD, the interface will always be consistent and placed in the virtual world where specified by the programmer.
2. From a more theoretical point of view, the interface also becomes one of the elements of the virtual world and can therefore be treated like any other object in the virtual scene. Elements of the HCI such as panels, icons and rulers, are treated just like any other 3D entities within the VE. Each element can be moved or scaled according to the user's needs. The user interacts with the objects through elements of this interface such as arrows which are placed

to help them edit the object, and feedback is provided through the visual modification of the same object in the scene.

The result of this approach is that any object in the virtual environment can be replaced, dragged or re-scaled at the convenience of the user, regardless of whether it is something they created or whether it was created by someone else present in the virtual world. In addition the same direct approach is followed for the interface which becomes just another part of the environment contributing towards an increase in the feeling of presence.

The 3D engine renders all the changes possible in the VE: movement of avatars, video conferencing streams rendered on 3D panels, textual communication through 3D chat, changing of the interface and most importantly, the creation and modification of objects created within JCAD-VR.

4.4.2 System Features

When JCAD-VR is initiated, the user is asked for a login name to be used to communicate within the virtual world, and through an options panel they decide which server to connect to and choose the server port (See Figure 4.5). In case no value is given the application can run in stand-alone mode.

The user can also decide whether to initiate the system in single or multiple screen modes. The former is provided for standard computer screens whilst the latter has been customised for the multi-projector Reality Center™ used throughout the research. Finally from the initial panel it is also possible to activate or de-activate the video conferencing facilities and if video conferencing is activated then support for video capturing device recognition and checking is provided.

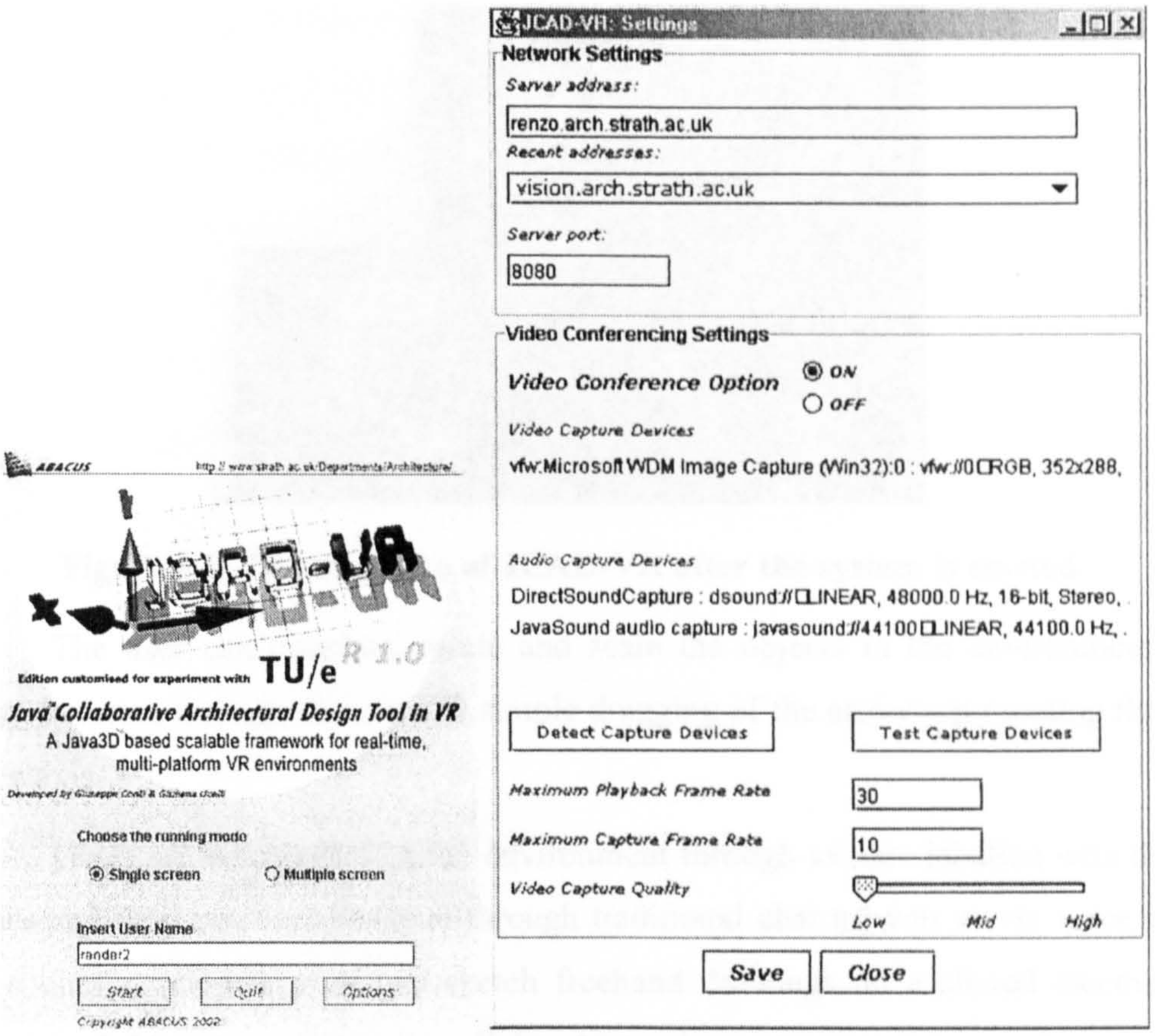


Figure 4.5: The initial panel and the option panel of JCAD-VR

Once the system is initialised every window disappears freeing the space for the 3D graphic user interface of the system and to the environment loaded during the start-up. A set of 3D menus and icons appears on the screen and through them each user can interact with the system and with the other participants (See Figure 4.6).

Through these menus it is possible to access a number of functions such as navigation and creation of objects. In the latter case a number of 3D shapes such as cones, boxes, spheres etc., and architectural objects such as walls, doors, windows etc., can be created and shared with other participants. An automatic procedure routinely checks for constraints and allows only the modifications that are possible. For instance, it will prevent a door being moved onto or too close to another door. A 3D-ruler and a 3D-panel close to the object constantly provide the user with feedback on the parameters that can be edited such as size, materials and cost.

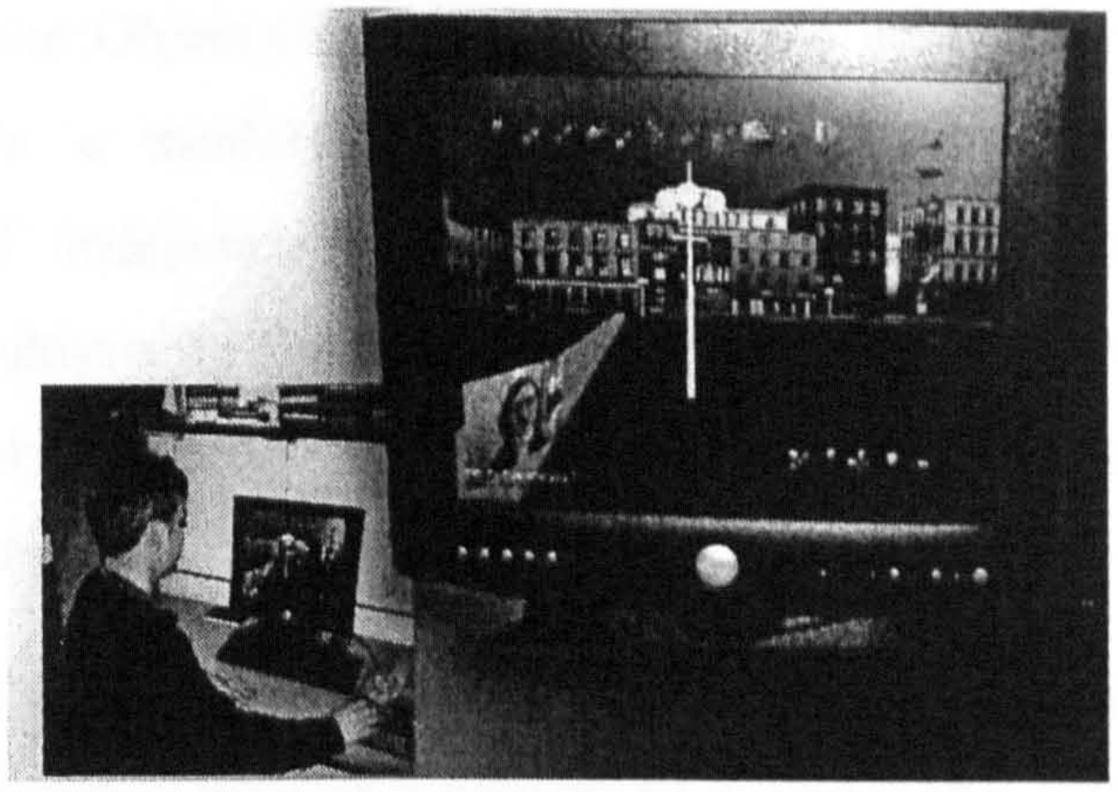


Figure 4.6: The interface of JCAD-VR after the system is started

The user can translate, rotate and scale the objects in the environment in every direction through the visually simple dragging of the arrow representing the x, y or z axis.

Users are represented in the environment through avatars labelled with their names and they can communicate through traditional chat as well as via voice and video conferencing or they can sketch freehand drawings on a shared electronic whiteboard. Moreover, due to its collaborative nature, any user can contribute to the creation and manipulation of the environment. Every time a user changes an object the system automatically upgrades this object's geometry and/or position throughout the network and the new configuration is made available to all the users in the world. This raises the issue of the possible concurrent action on the same object by more than one user. An object locking mechanism, implemented in JCAD-VR, tackles this problem by ensuring that every time an object has been selected it becomes inaccessible for any other user until it has been released.

Finally the system supports communication and interaction between users, it allows real synchronous collaborative design and it makes the design process a true multi-user collaborative experience.

4.4.3 The Modular Approach

The overall structure of the JCAD-VR system is shown in Figure 4.7. The system is divided into two different packages: a server and a client.

Following the Object-Oriented paradigm the architecture of JCAD-VR has been developed in a modular fashion. This has allowed scheduling of the implementation of independent self-functioning modules, concurrent software development and ultimately the delivery of a working prototype as a functioning core. Simultaneously this ensures the future expandability of the system through the implementation of new modules.

Each package is made of different units each containing one or more cores. Each core can be accessed through a number of modules that have a specific function within the main framework.

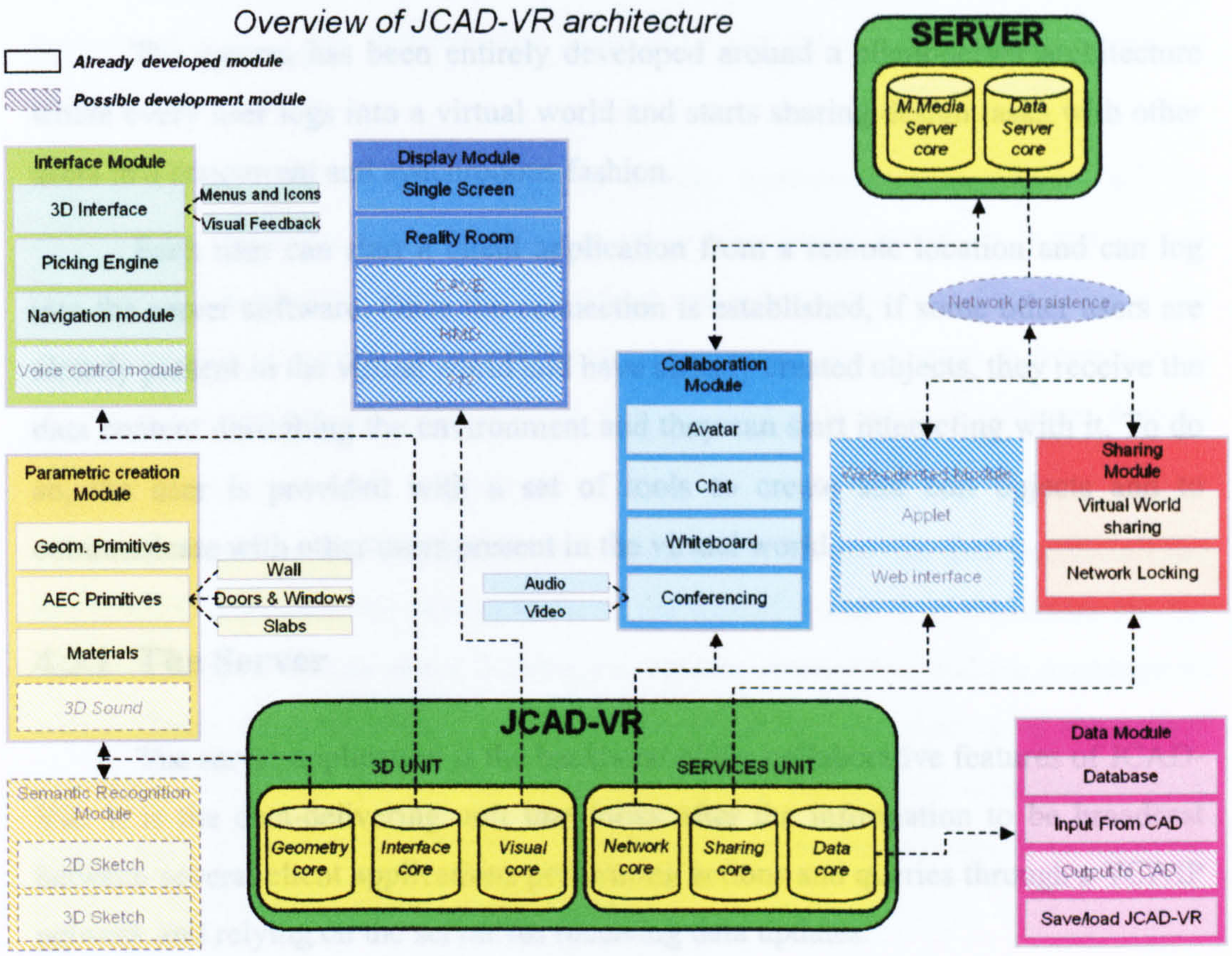


Figure 4.7: The JCAD-VR framework schema

The adoption of this approach gives a very high level of flexibility to the overall architecture since specific functions can be included at different times without interfering with the general structure of the framework. Rather than developing a monolithic closed design system, the modular approach provides ready-to-use sub-systems dealing with different functions. This way of working is therefore

less error-prone, since it allows the control of independent sub-sections, it strengthens the general structure of the system and it brings extreme flexibility and expandability to the entire system.

In the following paragraphs an overview of Figure 4.7 will be provided and the meaning of the different elements will be given. The set of hatched modules in the diagram represent modules not developed and of lower priority. They will be subject of the last chapter which discusses the conclusions and possible further developments of the system.

4.5 The Client-Server Architecture

The system has been entirely developed around a client-server architecture where every user logs into a virtual world and starts sharing design tasks with other users in a concurrent and synchronous fashion.

Each user can start a client application from a remote location and can log into the server software. Once the connection is established, if some other users are already present in the virtual world and have already created objects, they receive the data content describing the environment and they can start interacting with it. To do so, the user is provided with a set of tools to create and edit objects and to communicate with other users present in the virtual world.

4.5.1 The Server

The server application is the backbone of the collaborative features of JCAD-VR. It is the data-delivering unit that looks after the information to be broadcast between several client applications performing actions and queries through a TCP/IP network and relying on the server for receiving data updates.

The server application is made of two cores (See Figure 4.7): the *JCAD-VR Data Server* which looks after the VE information to be broadcast, and the *JCAD-VR Multimedia Server* which streams audio and video for the video-conferencing tool. Both parts are closely linked to each other and they are seamlessly integrated to comprise the general server package.

As an independent part of the framework the server has an autonomous and simpler interface that provides primarily information about the network status.

The intrinsic multi-platform nature of JCAD-VR, inherited from the language used to code it, allows the server to transmit data to a broad range of platforms, from normal PCs to a supercomputer running a Reality Center™.

4.5.2 The Client

The client is the application used to run the virtual environment. To do so two closely connected units have been developed (See Figure 4.7): a *3D unit* and a *services unit,* each made of three different *modules*. Using the analogy of the human body the former could represent the heart while the latter could be considered the nervous system.

The *3D unit* is the broad part of the framework that handles all the information regarding the "visible" aspects of the virtual world. It is the logical container of three cores: *geometry*, *interface* and *visual core*. The first includes the code necessary to create and modify geometric entities, the second to interact with the human computer interface and the third to deal with several different display devices.

The *services unit* instead is the section of the framework that handles all the information regarding the management of the virtual world. It is the centre for the interconnection between users: it manages network connections and the exchange of data between users through a *network core*, it handles the sharing of the virtual environment through the *sharing core* and finally, through the *database core*, it keeps track of the state of the virtual world and it makes the retrieval of information possible about objects present in the scene.

4.5.2.1 The Geometry Core

The *geometry core* handles the creation of 2D and 3D objects. It permits the retrieval of 3D-objects present in the library and it allows the creation of three-dimensional geometrical primitives such as cones, boxes and spheres. Il also allows the creation of more complex and specific architectural entities, referred to in the

following chapters as AEC (Architecture Engineering Construction) objects, for example walls, slabs, windows, and doors.

Even though further details will be provided in the following chapters it is worth mentioning the great difference between "simple" shapes and specific AEC objects. For instance, although a wall is essentially a box from the visualisation point of view the system treats it in a completely different way. In fact, while a box is just regarded as a simple shape without any extra qualities apart from low-level attributes like material or cost, the wall is considered as an entity with "topological" properties. First of all it is made of two different surfaces - the internal and external face - and of an internal core. Furthermore it can be the parent of another object - such as a window or a door - and therefore it can hold information of a different type such as the number of windows or doors attached to it and their relative position.

The geometry module also provides the means for attaching materials to objects. These materials are stored, together with a number of 3D objects, in a library available to the user.

4.5.2.2 Interface Core

The *interface core*, as its name suggests, handles the Human-Computer Interface. As previously mentioned one of the main goals of the JCAD-VR framework was to achieve a *transparent* interface. The concept of *transparency* refers to the idea of an interface that is not detached from the 3D world but on the contrary is an integral part of it. As mentioned, instead of using traditional menus and toolbars, this 3D interface allows the user when immersed in an environment to find the means for the interaction within the VE itself. Therefore the user can interact through 3D widgets present in the virtual world which can be manipulated at their convenience just like any other object in the environment.

Likewise visual feedback is also provided within the environment, for instance in the form of 3D-rulers showing the size of objects or 3D-icons showing the operation being performed on an object.

Due to the limited need for advanced settings, such as front or clipping distance, field of view etc., a traditional window-based control panel was provided to complete the interface for advanced visualization options.

4.5.2.3 Visual Core

The *visual core* is the part of the framework that allows the interface with the visualization devices through *display modules*. The obvious computational constraints imposed by the use of different hardware is solved by creating a structure that is flexibly scalable which can also deliver images for a range of viewing devices, from the simple desktop monitor to the more complex tessellated screen for immersive environments. When JCAD-VR is loaded the user is asked to choose whether to work on a single screen or in a multiple screen mode and they can switch between these two modes according to the machine the application is running on.

The adoption of a modular approach to the framework shows clearly its advantages in the development of these modules. In fact in response to the obvious hardware limits imposed by the use of different platforms, the system had to be written to be easily customised to run on PCs as well as on the Sgi supercomputer. In the former the system runs on PCs whose video-card displays the virtual world on a traditional window or at full screen while in the latter JCAD-VR can take advantage of the 12-processors Sgi Onyx2 system which power the Reality Center™, ABACUS, in the University of Strathclyde, Glasgow (University of Strathclyde, 2002).

The modular architecture of JCAD-VR allowed the development of independent modules that can be easily customised to respond to the needs of each specific hardware configuration. Therefore when JCAD-VR is launched on the Sgi the *display module* customised to handle the Reality Center™ can take advantage of the supercomputer's graphics power to run on a 5-metre wide 2-metre high, tessellated screen where 3 Barco projectors create a 160-degree panoramic image. In addition to this approach, other *display modules* might be easily adapted to allow use of different VR devices such as CAVEs or Head-Mounted Displays without interfering with the general structure of the system.

4.5.2.4 Network Core

The *network core* connects the independently coded client and server packages of the framework providing the means for the transmission of information through a communication channel based on a TCP/IP network.

To ensure communication between users, represented in the 3D world by avatars, different means are provided, from basic chat to voice and video conferencing. Freehand sketching in 2D is also possible through a shared electronic whiteboard. Every user can take advantage of the synchronous multimode communication media while working collaboratively on a design task.

4.5.2.5 Sharing Core

Like the *network core* the *sharing core* bonds clients and server through the exchange of numerical information. This part of the system broadcasts information about the objects present in the virtual environment and transmits actions and modifications performed on them thus providing much of the foundation for the collaborative features of the system.

This core ensures network consistency through a distributed network-locking mechanism that attributes a unique number, consistent for all the users in the system, to any object created across the network and sets the user priority on selected objects. When a user selects an object, this becomes *locked* and an event is sent through the network to other users to prevent them accessing the object. Locked objects can no longer be chosen by other participants until they are unlocked. This mechanism does not allow more than one user to edit an object at the same time and it is designed in order to ensure consistency throughout the system.

4.5.2.6 Database Core

Finally the *database core* handles the internal database that keeps track of the creation or manipulation of objects in the virtual scene by all the users present in the environment. Through it the system can retrieve information on geometric primitives and materials, etc. The internal database is closely coupled with the network core. It does not only keep track of what is happening within the user's virtual world but

most importantly, upgrades through the network, the information broadcast by other users' internal databases. If for instance a new object is created or its status is changed, the system will upgrade the internal database of each user in real-time no matter who is performing the action on the object.

Finally, for the convenience of the user, an I/O module allows storing and retrieval of the database content for the save/load operation.

4.6 Delimitation of Scope

The framework proposed in the previous pages tries to address the major issues of a collaborative VR-based design tool for architecture. Nevertheless a number of limitations narrow the scope of this framework, specifically:

- The implementation of the application is only at the prototype stage and it has to be considered as a proof of concept rather than a finished commercial package.
- In general the implementation of new tools was preferred and has been considered of greater scientific interest than the optimisation of already developed modules.
- The application has been specifically designed for architects and not for Virtual Reality experts therefore particular attention was paid to having a familiar graphical user interface and its ease of use.
- The research focuses on the early stages of the architectural design process thus the software application has been developed to be used during the conceptual modelling phase and not as an advanced modeller.
- JCAD-VR is a tool to promote flexibility in design not a CAD/CAAD package and therefore it lacks the routines and algorithms that can handle complex geometries and solid modelling tools usually provided in commercial applications. The abstraction required at the first stage of the design process justifies the support for elementary shapes.

- Although it has been shown to provide support for immersive and semi-immersive VR configurations, the implementation of the system as mainly a Desktop-VR application is justified by the collaborative nature of the system. The need to run several clients concurrently makes the Desktop-VR choice the only realistic approach for a feasible multi-user VR experience.

As already mentioned, due to the evident complexity of the framework proposed, the developed prototype represents a limited, yet fully functioning, part of the main schema represented in Figure 4.7. Therefore the result should be considered as proof of the feasibility of such an application for architecture.

The previously mentioned shaded modules of Figure 4.7 represent modules with a lower priority which have not been developed and their further development will be subject of the last chapter of the thesis. An order of priority was imposed on the modules in order to deliver the most important or innovative features of the framework. In this way the overall system was provided with a substantial set of functions addressing all the main issues mentioned in the research and making the final prototype a fully functioning application.

4.6.1 General Technical Issues

From the technical point of view, the approach followed when programming JCAD-VR has mostly favoured functionality or user-friendliness over efficiency, both in the code and in the overall performance of the system. The choice could be summarised by stating that programming paradigms, as well as coding solutions, have been chosen to privilege the user's point of view rather than the absolutely perfect performance of the system. The choice has been justified by the ever-increasing power of the hardware that has brought the overall system to a satisfactory level of performance on most systems, well above the threshold of 30 frames per second.

Further, the multi-platform nature of the Java™ programming language has raised the issue of performance across different platforms and operative systems. The wide range of hardware configuration available across the number of platforms virtually supported by the system has lead to a choice in the optimisation of the code.

Due to its wider availability, and its achievement of excellent rendering performance the Windows platform was considered of greater importance to the scope of the project. In contrast a lower priority was given to the optimisation of the code for the Sgi Irix platform, due to its range of hardware configurations, graphics subsystems and visualisation devices which would make it too limiting to use this particular configuration as the sole benchmark to evaluate the application.

The different ways the Java™ Virtual Machine (JVM™) handles threads in Unix-based systems, together with the thread-related issues typical of multi-processor systems and the inefficiency of the Irix port of the JVM™, specifically produced a lower, yet satisfactory, performance on the Sgi system used. In addition the system has not been fully tested using Linux and it has not been tested at all with the other supported operative systems such as Solaris, MacOs X, etc.

The fact of the always-increasing number of new releases of JVM™ and of the APIs used by JCAD-VR might contribute to incompatibility issues. Therefore Appendix D has been reserved to specify the hardware and software used to fully test the application.

4.7 Conclusions

The multidisciplinary nature of this research was the opportunity to investigate collaborative design issues, the role of interfaces inside CAAD packages, the design process in the first stage of its conception, the use of Virtual Reality in architecture as a design tool, the issue of collaborative systems and finally a number of technical issues.

As previously shown JCAD-VR proposes a user-centred prototype of a collaborative VR application where intuitiveness and control over the environment results in an innovative and more involving design experience.

The following chapters will detail the development of the system, and they will focus on the part of the framework depicted in Figure 4.8 and will provide an in-depth view of its features and the technical solutions.

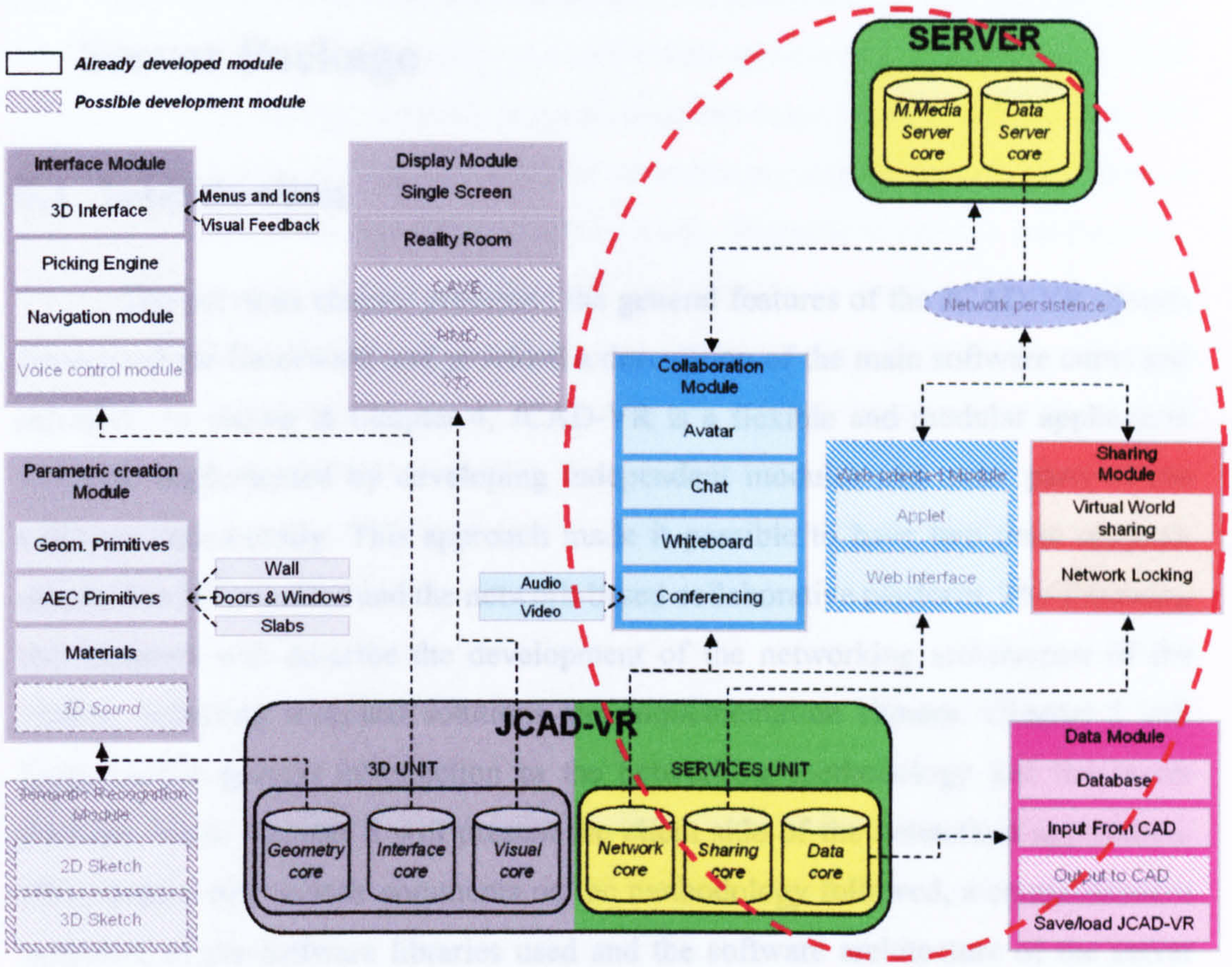


Figure 4.8: The portion of the framework studied in this thesis

5 General Methodology and Implementation of the Server Package

5.1 Introduction

The previous chapter presented the general features of the JCAD-VR system introduced the framework and provided a description of the main software cores and modules. As shown in Chapter 4, JCAD-VR is a flexible and modular application that was implemented by developing independent modules and other parts of the software concurrently. This approach made it possible to have two main research efforts: the 3D interface and the network-based collaborative platform. The following two chapters will describe the development of the networking architecture of the system, including technical solutions and implementation choices. Chapter 5 will focus on the general introduction to the networking methodology and the server package, while Chapter 6 will present the client side of the networked application. This chapter will include comments on the methodology followed, a comprehensive overview of the software libraries used and the software architecture of the server core of the system. Several diagrams will help in explaining in detail the technical solutions for developing the system and will help to minimise the need to examine the less understandable underlying source code. Hughes et al., (1997) inspired the graphic convention for these diagrams.

5.2 Justification of the Methodology and Technical Choices

JCAD-VR is an original application designed specifically to address the use of VR in collaborative design. This particular approach was preferred to the adoption of existing complex CVE platforms (See Section 3.4.4) covering all the requirements of setting up of a CVE but forcing the programmers to extend their system's features using their programming languages. In addition the development of the JCAD-VR application was the opportunity to build an original framework and to address theoretical as well as technical issues which were specific to the needs of architects.

The Collaborative unit of JCAD-VR, together with the rest of the software, was implemented entirely in Java™ language. The choice of a Java™-based network system was based on its suitability for networked applications programming and its portability. Historically, network programming has been difficult, error-prone, and complex due to the various layers of the networking protocol and the multitude of different functions in each library (Eckel, 2000). Procedures such as handshaking, connecting, packing and unpacking data and finally transmitting data over the network had to be individually handled by programmers. Java™ language allows the abstraction of the underlying form of the network making the implementation of an application more efficient and less time consuming. Also the core Application Programming Interface (API) of Java™ provides a number of standard *classes*, or data types which describe a set of objects having identical characteristics and behaviors (Eckel, 2000), and this allows access to networking protocols across a number of platforms. Since JCAD-VR is a collaborative VR application letting a number of clients from different remote locations work together on design tasks, it has to support flexibility in computer configurations and portability to the various platforms, which makes Java™ the most suitable language for implementing this networking architecture.

The collaborative unit of JCAD-VR is based on a server/client framework, which consists of a multithreaded server broadcasting data to a number of users, and on a client application that manages a world database, containing information about the VE while receiving updates of data from the server. The choice of a server/client architecture, instead of peer-to-peer connections, was mainly due to the degree of control that it permits over the whole application, especially in a multi-client context. A peer-to-peer approach requires a large number of connections to allow the multiple clients to transmit data among themselves, as it aims to establish communication with one computer at a time. Furthermore the server manages client connections and transmission in a centralized manner, with all the distinct components of the application communicating over a single communications channel (See Figure 5.1).

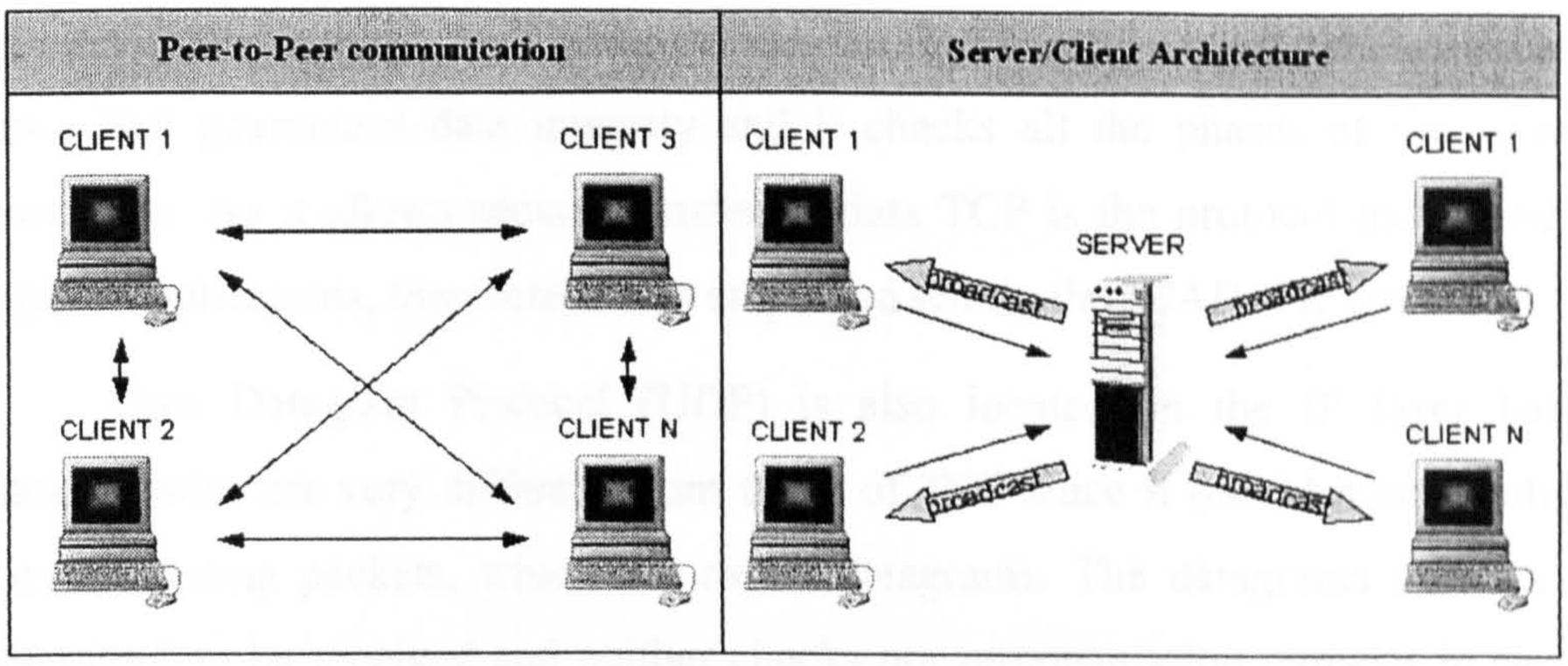


Figure 5.1: Peer-to-Peer networking architecture versus Server/Client architecture

On the use of a client/server approach for distributed applications, Macedonia et al. (1994) suggest that in dynamic large scale worlds servers can become "bottlenecks" for the application increasing the inherent latency of the virtual environment. This becomes crucial in large distributed virtual applications, such as military simulators described in Chapter 3 (See Section 3.4.1). By contrast JCAD-VR is a small scale multi-user application, it has been designed for relatively small groups of users and since each client holds data about the virtual world locally, the server broadcasts much smaller messages to update their databases, resulting in a relatively use of the underlying networking channel.

In choosing the communication channel that eventually allows transmission of data through various transport protocols with various characteristics, the TCP/IP approach was adopted for the general transmission of data and the UDP protocol was restricted to the transmission of audio and video. The TCP/IP infrastructure is organized on several levels, in which the lowest is the Internet Protocol (IP). IP handles addressing and routing and its packets can be sent over different media, such as the Ethernet, telephone lines, or by fibre optics. At the upper level of the infrastructure is the Transmission Control Protocol (TCP) that allows the reliable transmission of byte streams through which software applications can establish the connection and send and receive data. TCP handles the task of breaking the stream of data into smaller portions before being sent, operates checksums to ensure no data is missing, sorts received packets ensuring their correct order, performs

retransmission requests for missing packets and has control over the communication flow. TCP guarantees data integrity and it checks all the phases of the network connection. As it allows secure transfer of data TCP is the protocol most used for Internet applications, therefore it was employed within the JCAD-VR prototype.

User Datagram Protocol (UDP) is also located on the IP layer but its characteristics are very different from those of TCP since it provides an unreliable way of sending packets, which are called datagrams. The datagrams sent are not guaranteed to be received and neither checks nor retransmission requests in case of missing of data are performed. Although UDP permits only this unreliable form of transmission of data, if compared with the other protocol, it has some advantages when dealing with significant quantities of data, such as in the case of the real time transmission of audio and video. TCP therefore performs all the controlling and checking of data flow adding considerably to the network transmission while UDP packets do not contribute to the congestion. If problems arise some packets would simply be droped from the transmission. The use of UDP is limited to self-contained transmissions of data where no assumption of the receipt of previous data is required because in the case of missing datagrams consistency in the data received is still assured. Because of these characteristics UDP is often used when significant network traffic and large amount of transmitted data are involved, such as in the case of audio and video streaming for the videoconferencing in the JCAD-VR application.

5.3 Overview of the APIs used

The collaborative unit of JCAD-VR was implemented using the standard API of Java™ Software Development Kit (SDK) (See Sections 4.3.1 and 5.2), the Prominence library and the Java™ Media Framework (JMF™) API.

The Prominence library is a network specific collection of classes, written in Java™, that include a generic server/client framework for collaborative applications (Hughes et al., 1997). This library provided a flexible collaborative platform that was at the base of the development of the server/client architecture of JCAD-VR.

JMF™ was used in this case to develop the video-conferencing module of the JCAD-VR application, since it provides a unified architecture and messaging

protocol for managing the acquisition, processing, and delivery of time-based media data such as audio and video streams (JMF™, 2002).

The Prominence library and the JMF™ API will be described in more detail in the following sections.

5.3.1 Prominence Library

The standard Java™ API provides several classes for general streaming of data but these are not readily available to perform the higher level of operations common to server/client communications. Prominence Dot Com, a company that develops and markets Java™-based Internet products designed the Prominence library to facilitate the development of server/client architectures for collaborative applications in the Java™ environment (Hughes et al., 1997). This provides the programmer with filter streams to ease advanced networking issues, such as the handling of message headers, of placing stream messages into queues for later processing, of routing messages to named recipients and it was especially useful multiplexing and demultiplexing capabilities. In addition the Prominence library provides a readily available and easily extendible server/client framework using this new library of network streams.

The next three sections will describe the qualities of message streams data encapsulation, of multiplexing and demultiplexing message streams and it will introduce the general server/client framework included in the Prominence library.

5.3.1.1 Message Streams

Many applications communicate through "conceptual packets" (Hughes et al., 1997, p. 230) where the server and the transmitted data can be seen as a "self-contained unit" (Hughes et al., 1997, p. 230). In other words many applications do not need a byte-stream communication channel such as the one provided by TCP (See Section 5.2), but they require self-contained independent data packets carrying information of their origin. This approach is essential when the server has to deal with many client applications concurrently and since it receives data from different

sources, using standard streaming classes, the risk of mixing data from different messages is increased (See Figure 5.2).

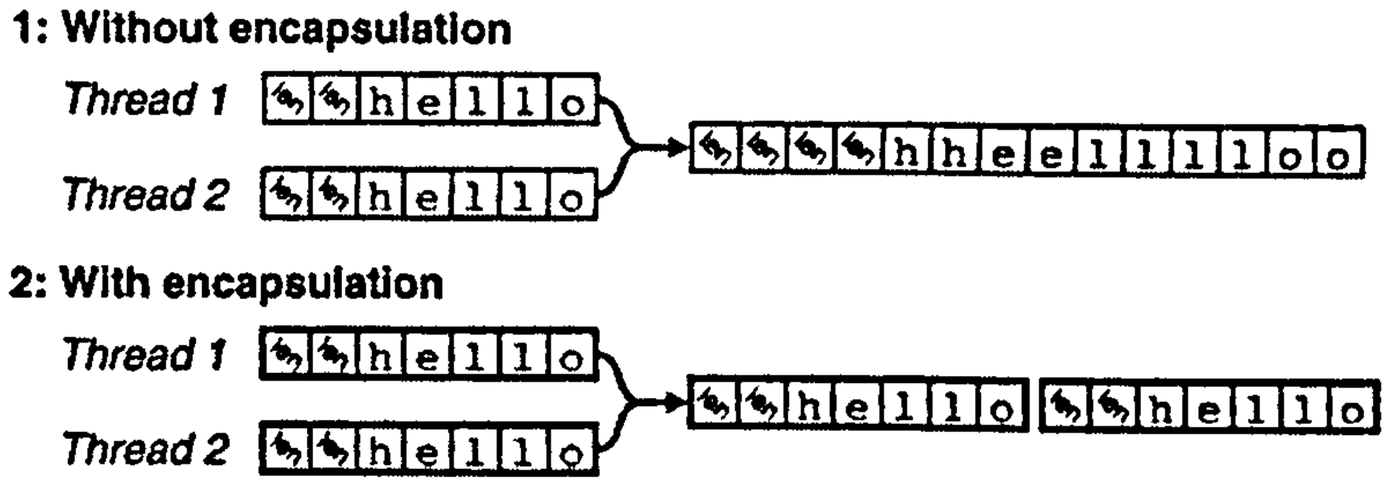


Figure 5.2: Encapsulation of discrete packets of information minimises the risk of mixing data from different data sources (Hughes et al., 1997, p. 230)

UDP (See Section 5.2) provides a packet-based transport mechanism but unlike TCP, it does not guarantee the recipient receives the data content. Therefore because in TCP the creation of packets is not automatic, it has to be performed *manually* specifying a unit of information, or packet, to be sent and it is necessary to assure that these are sent separately. Message streams, developed in Prominence library, allow message-based communication of arbitrary size and data format for complex multi-user networked applications. Further, the encapsulation of messages into discrete packets rather than just transmitting a continuous stream of data brings significant advantages to the communication. Advantages include the possibility to transmit several distinct messages using a single communication channel and the fact that errors in a single message do not spread to the subsequent messages (See Figure 5.3). In addition, without encapsulation, if we assume that every message is formatted in the same way, in the case of wrong data formats the system cannot solve the problem and so it spreads to the subsequent messages affecting the entire data transmission. While using encapsulation, since messages are extracted separately, the problem is limited to only the message with the wrong data format.

Finally encapsulation also hides network failures and assures data consistency, for instance in the case of network problems during the transmission of a message, the received part of the message will simply be discarded without causing the interrupted delivery of the rest of the message.

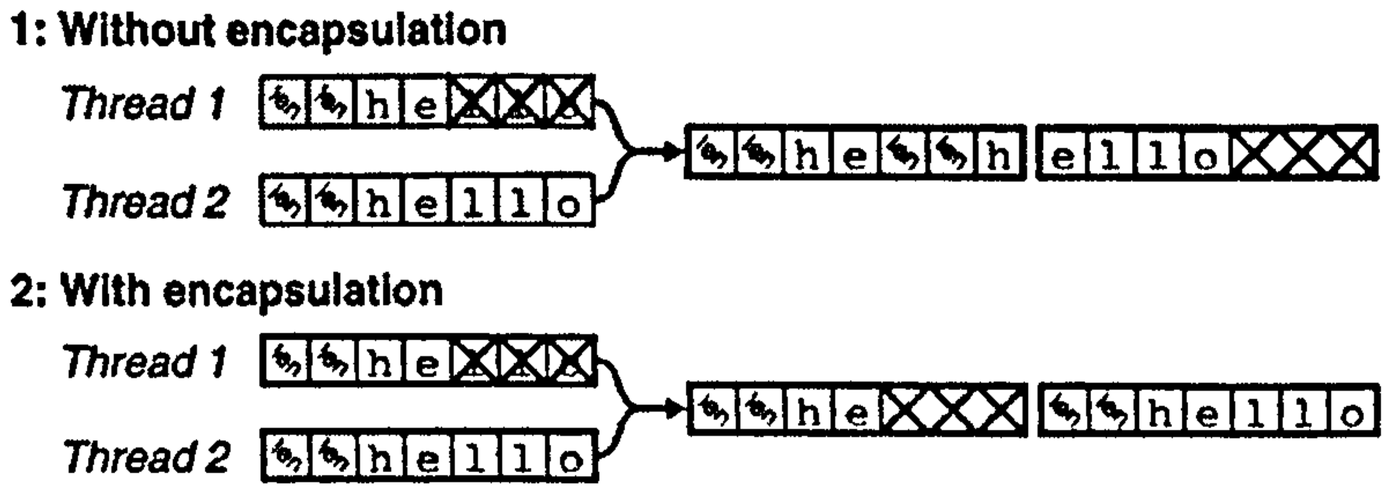


Figure 5.3: Encapsulation prevents errors in particular messages affecting subsequent messages thus minimising problems in networking communication (Hughes et al., 1997, p. 231)

The Prominence library provides encapsulation by introducing the *abstract* classes *MessageOutput* and *MessageInput* streams (Hughes et al., 1997, p. 232), and then by implementing specific classes according to the particular communication needs. An abstract class, in Java™ language, is an empty class that defines its general internal architecture, including the declaration of all its internal routines, or *methods*, without providing exact definitions of the methods functions. In other words it is a common interface that can be expressed differently for each subtype that *extends* it, but it does not provide any particular implementation. It also defines abstract methods that are incomplete by having only "declarations but not method body" (Eckel, 2000, p. 326) which means that they are empty.

Therefore *MessageOutput* is a superclass of Prominence message output streams and it extends the standard *DataOutputStream* class of Java™ by including all its commonly required methods while adding some completely new ones. *MessageOutput* operates encapsulation of messages into packets by attaching a header to the message body thus, as Hughes et al., (1997, p. 232) define it, it is "an encapsulation wrapper for a *DataOutputStream*". In practise messages are buffered until the call to its *send* method is performed and then this attaches the header, sending the message and resetting the buffer (See Figure 5.4).

The header can be of different formats, depending on the particular subtype, and it provides information for the extraction of messages from the data packets. For instance a simple implementation, the header carries information on the length of the

message body or, in a more sophisticated implementation such as in multiplexed streams, it transmits information on the application that originated the data.

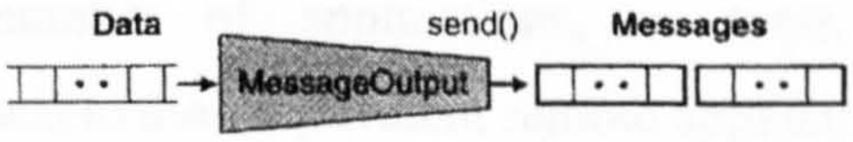


Figure 5.4: MessageOutput encapsulates messages into packets through its send method (Hughes et al., 1997, p. 232)

While *MessageOutput* provides a common interface for message output streams, *MessageInput* is a superclass for all the input message streams of Prominence library. A message input stream performs decapsulation of messages from the network stream and presents the extracted data to an application. It extends the standard Java™ *DataInputStream* class by adding a *receive* routine to the usual methods of this class (See Figure 5.5).

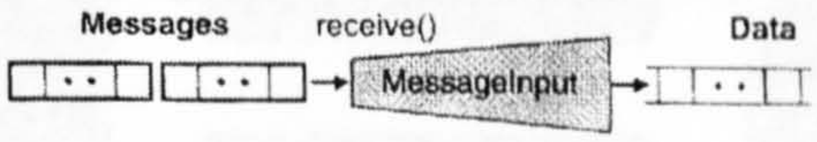


Figure 5.5: MessageInput decapsulates messages from data packets through its receive method (Hughes et al., 1997, p. 234)

This new method waits "until a complete message has been received, unwraps the body of the message from the header, and then makes the message body available for reading" (Hughes et al., 1997, p. 233). In effect, the *receive* method receives a packet from the stream, opens it and makes its content available for the application.

As described *MessageOutput* and *MessageInput* cannot be used but they have to be implemented in some form in order to fill their empty methods of specific functions. Within the Prominence library there are a number of specific implementations including classes allowing the queuing of messages, routing and most importantly allowing multiplexing mechanism. The next section will describe in detail multiplexing and demultiplexing streams and will provide basic information on the management of networking for concurrent multi-application software.

5.3.1.2 Multiplexing and Demultiplexing Message Streams

In general collaborative software allowing concurrent work involves the development of a number of applications, or tools, that are linked via communications channels to their equivalent remote applications. For instance, every time a participant is working collaboratively using a tool, their actions are notified to all the remote clients by modifications to the tools of the same type. In practise every user's action is able to send a certain type of data convenient for that kind of tool, through a communication channel, while the same tool, at remote locations, is ready to receive data of a suitable typology carrying information about remote users' actions (See Figure 5.6). Once the data has been received its content is translated into a change of status of the tool and the remote users' actions are then visualized.

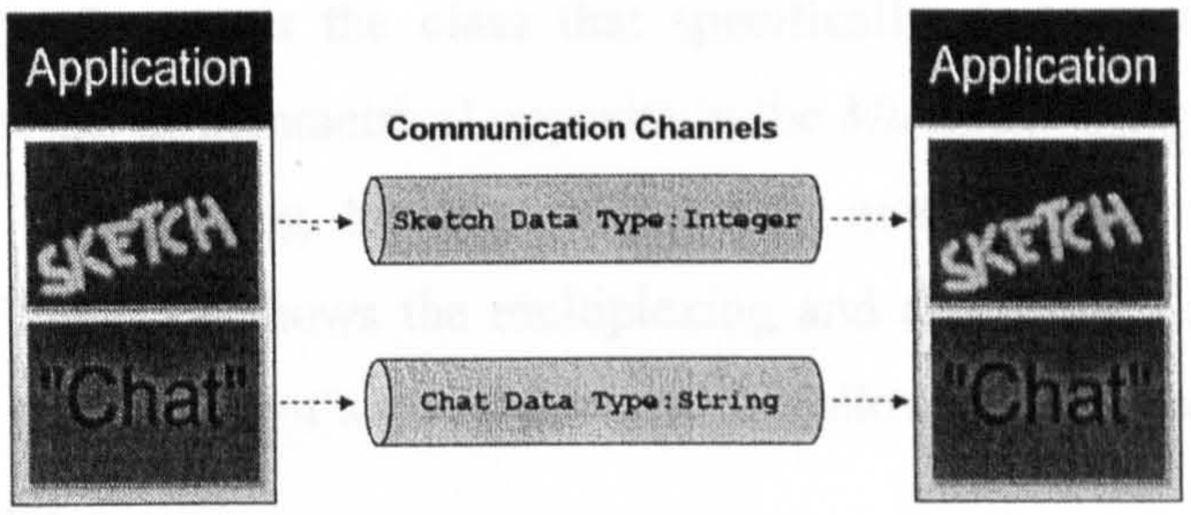


Figure 5.6: Communication among remote applications within a collaborative software involves sending and receiving of predefined types of data through communication channels

As shown Prominence library offers a number of network streams with more advanced capabilities than the standard Java™ classes. Among these are the multiplexing and demultiplexing message streams that extend the *MessageOutput* and *MessageInput* superclasses of Section 5.3.1.1 bringing significant advantages to the standard classes allowing the development of complex networked applications.

In the Collins concise English dictionary (1982, p. 740) *multiplex* means "The use of a common communications channel for sending two or more messages or signals". The purpose of multiplexing is therefore in order to enable several independent components, applications or tools of a client to communicate through a single communication channel (See Figure 5.7).

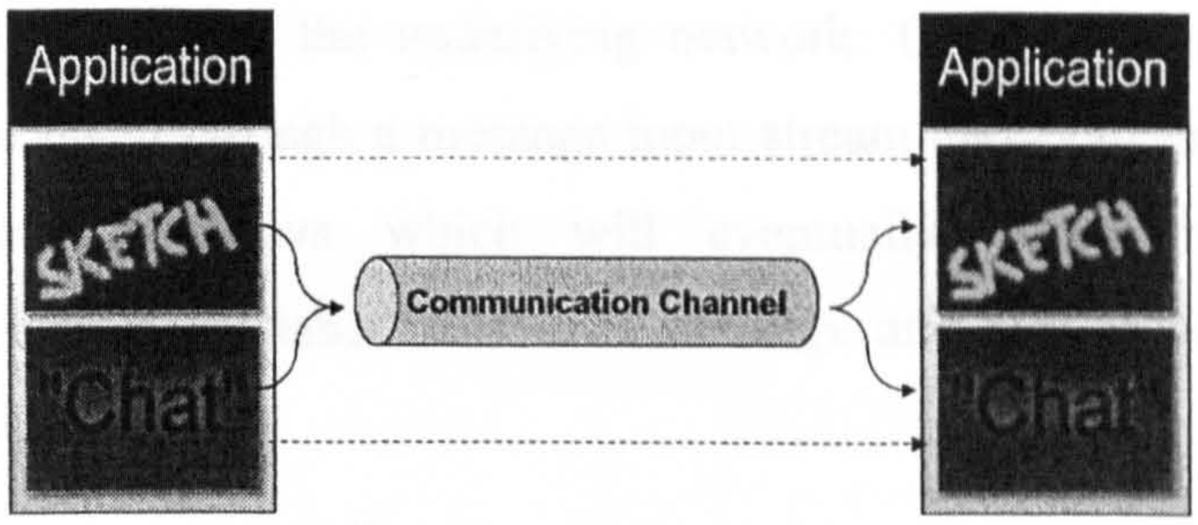


Figure 5.7: Multiplexing allows the connection of two or more tools using the same communication channel

In particular, multiplexing is based on the concept of adding a header, or label, specifying its origin at the beginning of every message before sending it to the communication channel through a *MessageOutput* stream. The *MultiplexingOutputStream* is the class that specifically deals with this labelling mechanism and it has its symmetrical opposite in the *MultiplexingInputStream* class, which extracts multiplexing headers from each message received through a *MessageInput*. Figure 5.8 shows the multiplexing and demultiplexing routines and depicts the path that the client's messages have to follow in order to be received by remote participants.

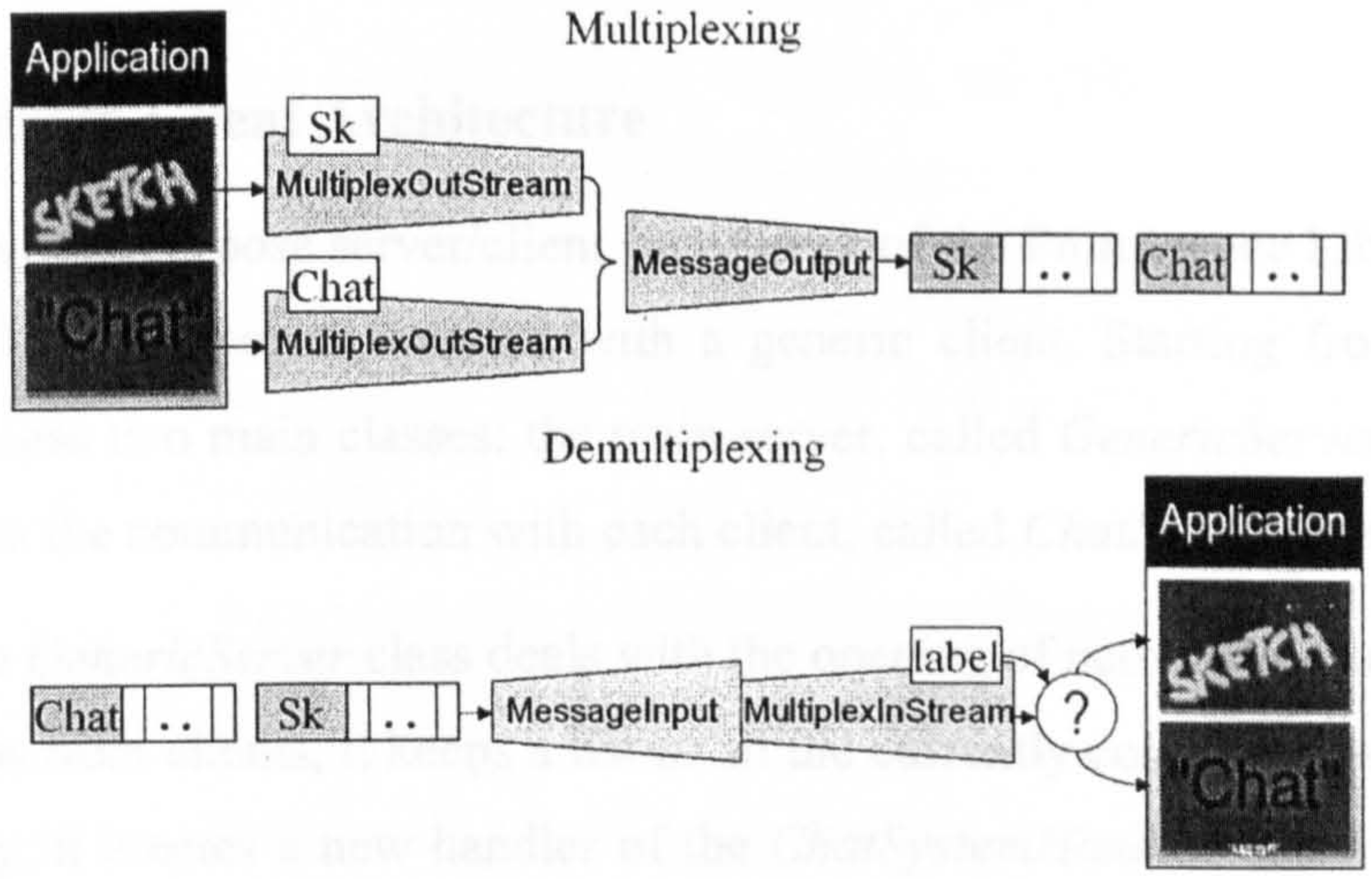


Figure 5.8: Multiplexing and Demultiplexing processes are based on the insertion of headers at the beginning of the packets to be streamed

Messages from each of the client's applications, shown in the figure as a sketch application and a chat board, have arbitrary application-specific labels added, *SK* for the sketch and *Chat* for the chat board, and then they are sent through a

message output stream to the underlying network. On the recipient side, these messages are received through a message input stream class and they are passed to the *MultiplexingInputStream* which will eventually, when coupled with a demultiplexer, extract the label from each message and deliver the content to the appropriate tool.

In conclusion, by following these simple steps, the multiplexing and demultiplexing classes allow data, originated by multiple tools, to be streamed through a unique communications channel offering increased flexibility to the network.

The Prominence library, apart from the message streams just discussed, offers a complete multipurpose server/client framework based on the multiplexing approach. This framework makes the management of the streaming of data significantly easier especially when the software is designed to allow many types of collaboration through a number of tools. The following section will examine this by outlining the general architecture of the server/client framework included in Prominence library, therefore providing a general description of the basis of the JCAD-VR networking system that originated from it.

5.3.1.3 Server-Client Architecture

The multipurpose server/client framework of the Prominence Library is based on a multithreaded server coupled with a generic client. Starting from the server package, it has two main classes: the main server, called *GenericServer*, and a class dedicated to the communication with each client, called *ChatSystemHandler*.

The *GenericServer* class deals with the opening of network sockets, it accepts connections from clients, it keeps a list of all the currently connected users and most importantly, it creates a new handler of the *ChatSystemHandler* class for each new client. This handler is then responsible for dealing with a particular client and it has the specific role of processing all its communications. Figure 5.9 shows the general architecture of the multithreaded *GenericServer* with the communications, which have been already established, conducted through handlers to the

ChatSystemHandler class. The connection of a new client makes the *GenericServer* create a new handler and start an independent process.

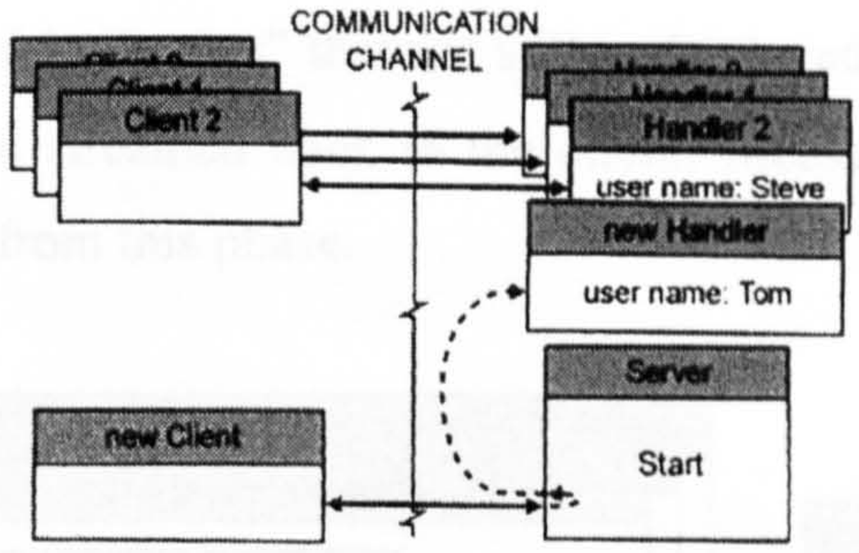


Figure 5.9: The GenericServer framework, image adapted from Hughes et al., (1997, p. 289)

The *ChatSystemHandler* class has the important role of processing client connections, separately from the main thread used by the server, hence it forms the basis of the multithreaded structure of the server. In addition every new client connecting to the server sends a username that is checked by the *ChatSystemHandler* in order to verify if it is already in use. Once the checking is successfully completed, the communication proceeds using the message streams discussed in Section 5.3.1.1 and 5.3.1.2. Figure 5.10 shows the internal architecture of the *ChatSystemHandler* class and the following list will help in clarifying the path that all the messages follow within the class before being sent to the client:

1. The *RoutingInputStream*, which allows the routing of messages to specified clients, receives as input an array of possible final targets from the multiplexed stream sent by the client.

2. The message is then passed to a *MultiplexOutputStream* that attaches a header giving the name of the client that sent the message.

3. The message is temporarily stored in a *hashtable*, titled "routes" in Figure 5.10 that is a bidimensional data container linking a content to a *key* identifying it, in this case the clients' names. This hashtable is shared among all the handlers of the *ChatSystemHandler*. In practise the hashtable represents the repository of all the communication content between the clients and the server. In short, "routes" stores the data content transmitted by all the clients sorted in temporal order, together with the client's name that sent the message. This hashtable is also

synchronized in order to prevent concurrent access to its contents by more than one handler at a time.

4. The messages stored in "routes" that are still multiplexed, are finally sent to the queue waiting to be streamed back to the client. Messages transmitted by the client are excluded from this phase.

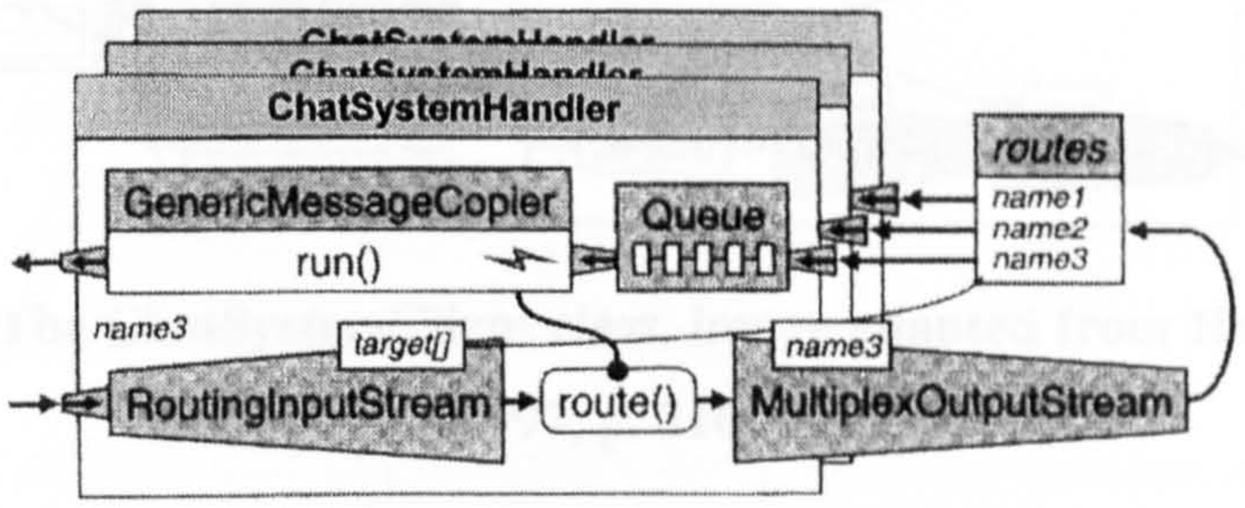


Figure 5.10: The ChatSystemHandler class, image adapted from Hughes et al., (1997, p. 315)

Each client's handler autonomously follows this procedure only sharing with the other clients the hashtable that holds the entire contents of the communication between server and clients. Note that the communication is kept generalised and the server is not aware of the exact content of the information since demultiplexing is never performed within the server package.

Finally *ChatSystemHandler* is responsible for performing the transmission to the joining participants of a list containing all the currently connected users and it announces joining or leaving clients.

Included in the client side of the Prominence framework, is the *ChatSystemClient* class that performs the network and stream set-up for those clients wishing to communicate with the *GenericServer*, and also the *ChatClient interface,* or the client's tools base class, that must be implemented by any client's tools wishing to join the server/client architecture.

The *ChatSystemClient* class will first of all open a socket connection with the server and then attempts to register the client's username. Once the server has

successfully performed the username registration, the *ChatSystemClient* class creates communication streams for the specified client's tools.

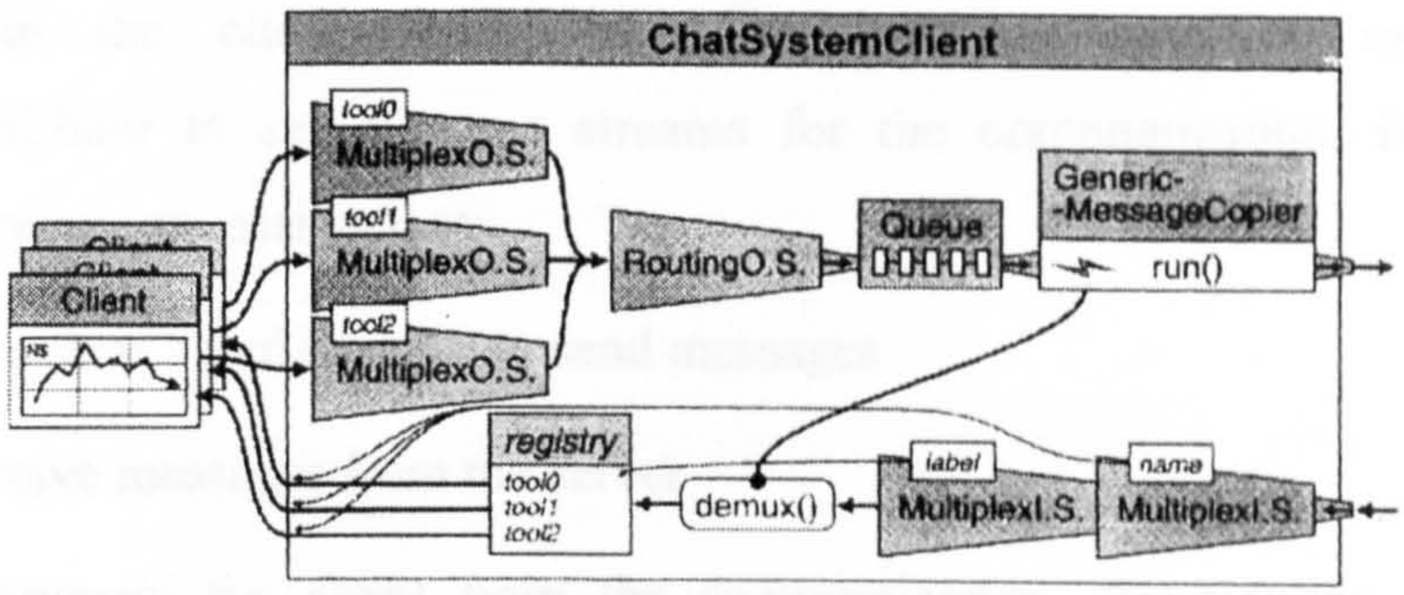


Figure 5.11: The ChatSystemClient class, image adapted from Hughes et al., (1997, p. 316)

Figure 5.11 shows the path that messages have to follow from each client's tool to the network and vice versa and the following list describes it step by step:

1. A hashtable, called "registry" in the figure, is created to map multiplex labels to registered client tools and each tool receives a *MultiplexOutputStream* to communicate.
2. Multiplexed messages are wrapped in *RoutingOutputStream* messages thus allowing the client to route them to specific users.
3. A queue temporarily stores the messages before being delivered.
4. The messages are finally sent to the server using a routine called *GenericMessageCopier* in the figure. It copies messages from the queue and sends them out to the network through a *MessageOutput* stream.
5. Concurrently a *MultiplexInputStream* receives incoming messages from the server, extracts the initial header with the client's name and then passes the packet to another *MultiplexInputStream*. Coupled with a demultiplexer this strips the second header indicating the tool.
6. The crude message and the label indicating the tool are now checked against the "registry" hashtable, which holds a list of all the client's tools, and the message is finally sent to the appropriate application.

All the client's tools must *implement* the *ChatClient interface* in order to be included in the client framework. The interface specifies methods for *ChatSystemClient* to set message streams for the communication. In particular message streams are assigned to:

1. The client's tool in order to send messages
2. Receive messages from the server
3. Disconnect the client from the communication, this implies ceasing the communication over the assigned message stream.

The *ChatClient* interface therefore deals with the management of all the incoming and outcoming messages from the tool, without showing the underlying architecture of the *ChatSystemClient* class.

Figure 5.12 and Figure 5.13 show the overall server/client architecture and the use of different types of message streams according to the needs of both the server and client.

The use of *GenericServer* and the relative client minimises the need to handle low-level networking issues for the implementation of collaborative applications. In addition Prominence library provides the programmer with a full range of classes that makes the development of networked applications easier and allows the effort to be focused only on the development and optimisation of the features of the shared applications and their related issues.

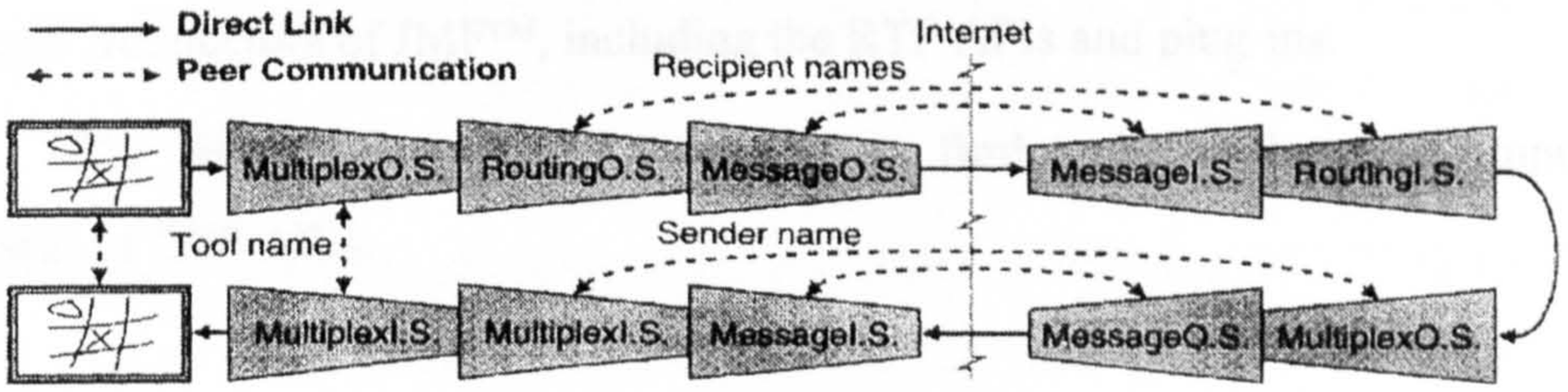


Figure 5.12: Use of several filtered streams in the server/client architecture (Hughes et al., 1997, p. 314)

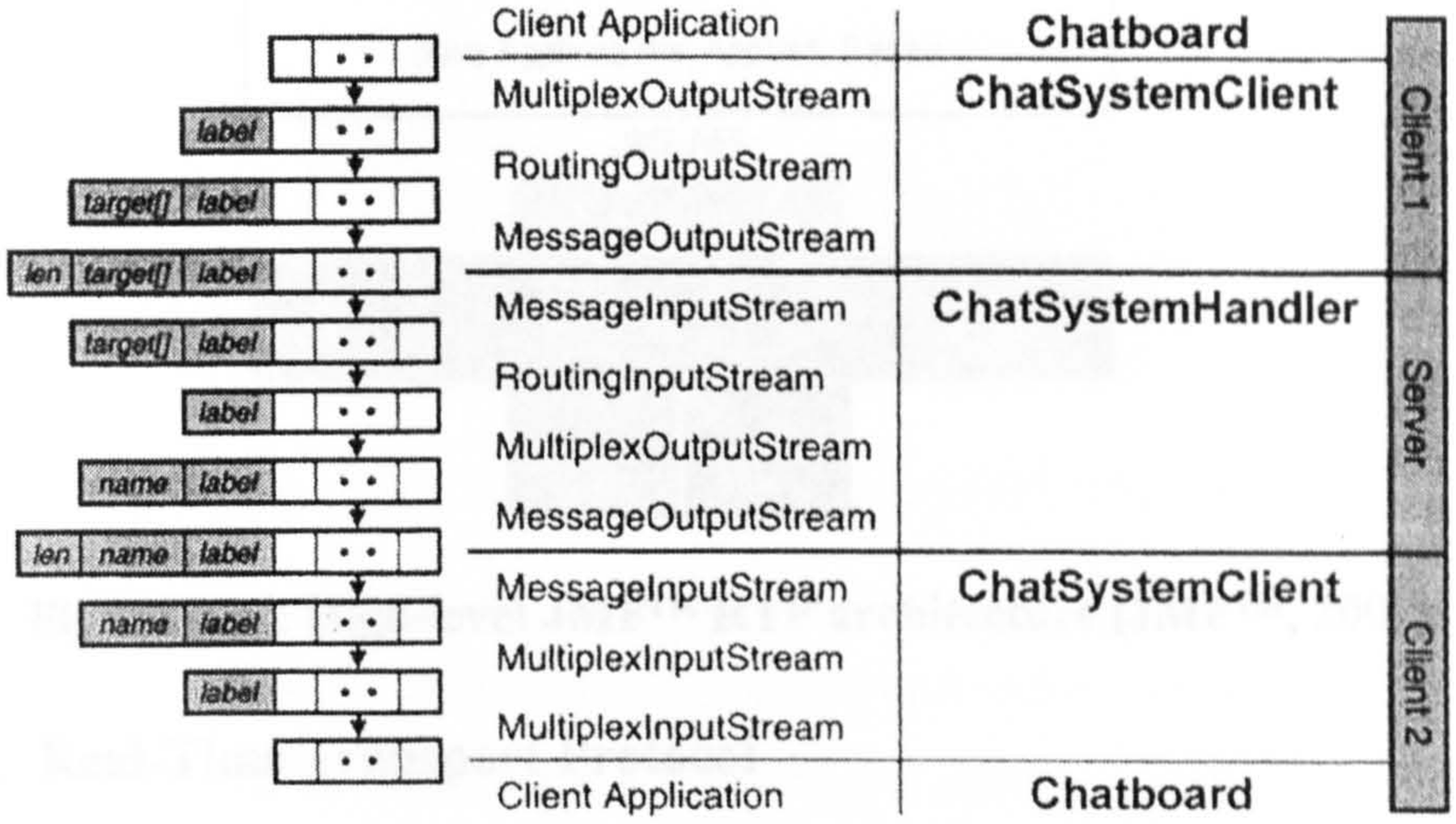


Figure 5.13: The sequence of various message streams in the server/client framework, image adapted from Hughes et al., (1997, p. 314)

5.3.2 Java™ Media Framework API

JMF™ 2.0 is an additional package of Java™ 2 SDK designed by Sun Microsystems, Inc. and IBM Corporation, which provides the user with a ready to use library for the deliver and rendering of synchronized data. It enables audio, video and other time-based media to be added to Java™ applications and applets, allowing the capturing, playback, streaming and transcode of multiple media formats. In particular JMF™ includes dedicated APIs for accessing and managing the Real-Time Transport Protocol (RTP). This enables the transmission and reception of real-time media streams across the network and the implementation of conferencing applications in Java™. Figure 5.14 shows the layer structure of the APIs and the general architecture of JMF™, including the RTP APIs and plug-ins.

The following section will discuss in further detail the most important features of RTP APIs.

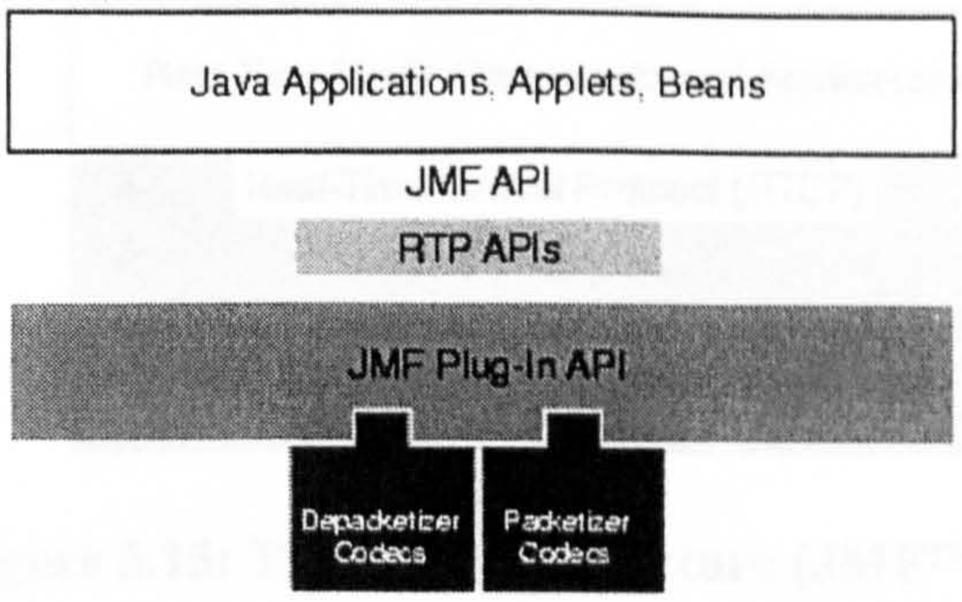


Figure 5.14: High-level JMF™ RTP architecture (JMF™, 2002)

5.3.2.1 Real-Time Transport Protocol

An important aspect of streaming real-time multimedia data over the network is that clients must be able to play the received media streams while the process of downloading is still in progress and a stream might not even have a predefined duration. Real-time transmission will also very often require significant use of network bandwidth. The approach used for streaming real-time data therefore will differ from the one used for transmitting *static* data, such as files, where the concern is mainly whether all the data has arrived at its final destination. In addition real-time transmission gives priority to the speed of the streaming regardless of any eventual loss of data. These specific requirements are achieved by the use of different networking protocols other than the usual TCP (See Section 5.2). The RTP protocol, a product of the AVT working group of the Internet Engineering Task Force (IETF) (RTP, 2002), is now the Internet standard for transporting real-time data such as audio and video. RTP is network and transport-protocol independent but it is often used over UDP (See Section 5.2) since the latter performs less checking on the data achieving higher transmission rates. Data packets sent via RTP are not guaranteed to be received therefore the receiver has to perform checks for data loss and has to reconstruct the correct order of the data packets. To make these tasks easier RTP is augmented by a control protocol, called RTP Control Protocol (RTCP) (RTP, 2002), which allows monitoring of the quality of the data distribution, permits retrieval of information about the data source and provides statistics on the transmission of data.

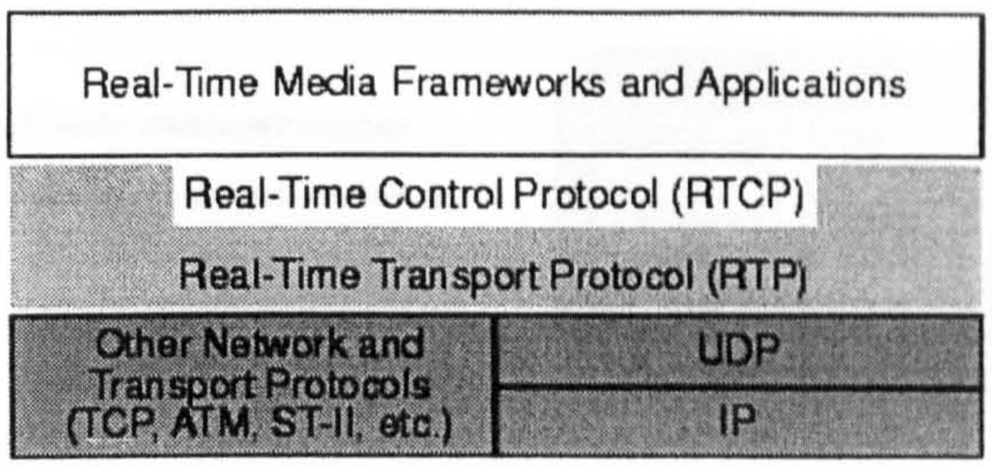


Figure 5.15: The RTP architecture (JMF™, 2002)

Finally RTP synchronizes media streams from different data sources thus allowing for instance the transmission of audio and video that have been captured from live sources of data such as microphones and digital cameras. This last feature makes RTP the perfect protocol for streaming data for video-conferencing applications.

5.4 The Network Package and its Implementation

Chapter 4 introduced the JCAD-VR framework and offered a general overview of the JCAD-VR networking architecture and its main features. The last sections of this chapter and all of chapter 6 will discuss in detail its implementation also in conjunction with the APIs and libraries described in Section 5.3.1 and Section 5.3.2.

Figure 5.16 illustrates the part of the JCAD-VR framework relating to the networking architecture and it shows the separate features of both the server and client packages. This section and Chapter 6 will present the server side of the system first and then the client application.

In general terms the JCAD-VR networking system extents and customises the server/client framework included in the Prominence library. Its low-level architecture is based on the Prominence library but JCAD-VR adds several original features to it and includes a networking architecture dedicated to video conferencing based on the JMF™ API.

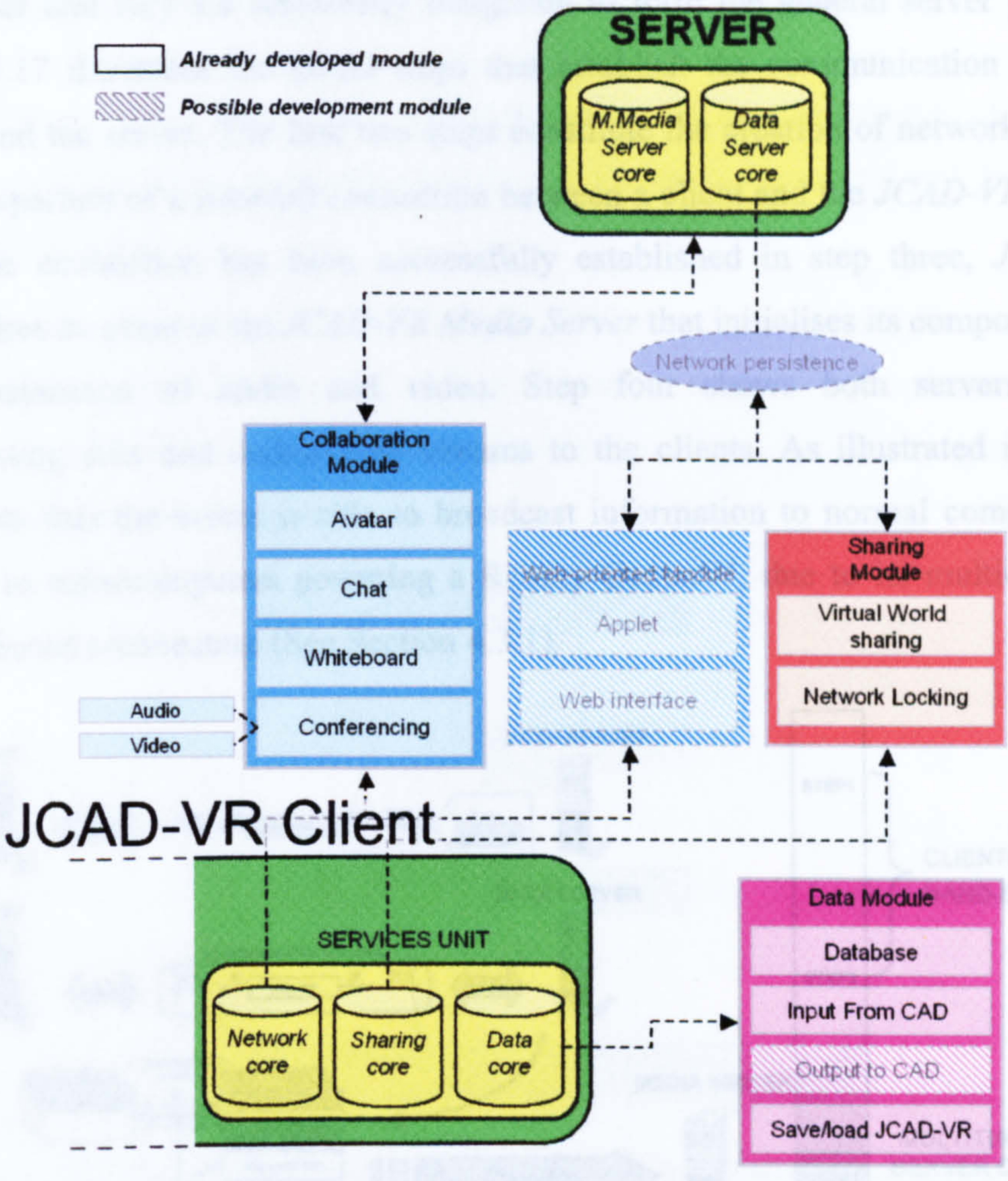


Figure 5.16: The JCAD_VR networking system extracted from the general JCAD-VR framework

The following section will offer a comprehensive explanation of the server package. To make the description easier its architecture will be illustrated through a series of graphs.

5.5 Implementation of the Server Core

Figure 5.16 shows the server and its components: a data server, called *JCAD-VR Server*, and a multimedia server, called *JCAD-VR Media Server*. *JCAD-VR Server* deals with the management of all the client connections, together with the transmission of geometrical information and it broadcasts data for the collaborative tools. On the multimedia side, the *JCAD-VR Media Server* handles the streaming of audio and video for the video-conferencing tool. Both servers are closely linked to

each other and they are seamlessly integrated to form the general server package. Figure 5.17 illustrates the initial steps that establish the communication between clients and the server. The first two steps constitute the creation of network sockets and the aperture of a network connection between a client and the *JCAD-VR Server*. Once the connection has been successfully established in step three, *JCAD-VR Server* fires an event to the *JCAD-VR Media Server* that initialises its components for the transmission of audio and video. Step four shows both servers finally broadcasting data and audio/video streams to the clients. As illustrated in Figure 5.17, note that the server is able to broadcast information to normal computers as well as to supercomputers powering a Reality Center™, due to its multi-platform Java™-based architecture (See Section 4.3.1).

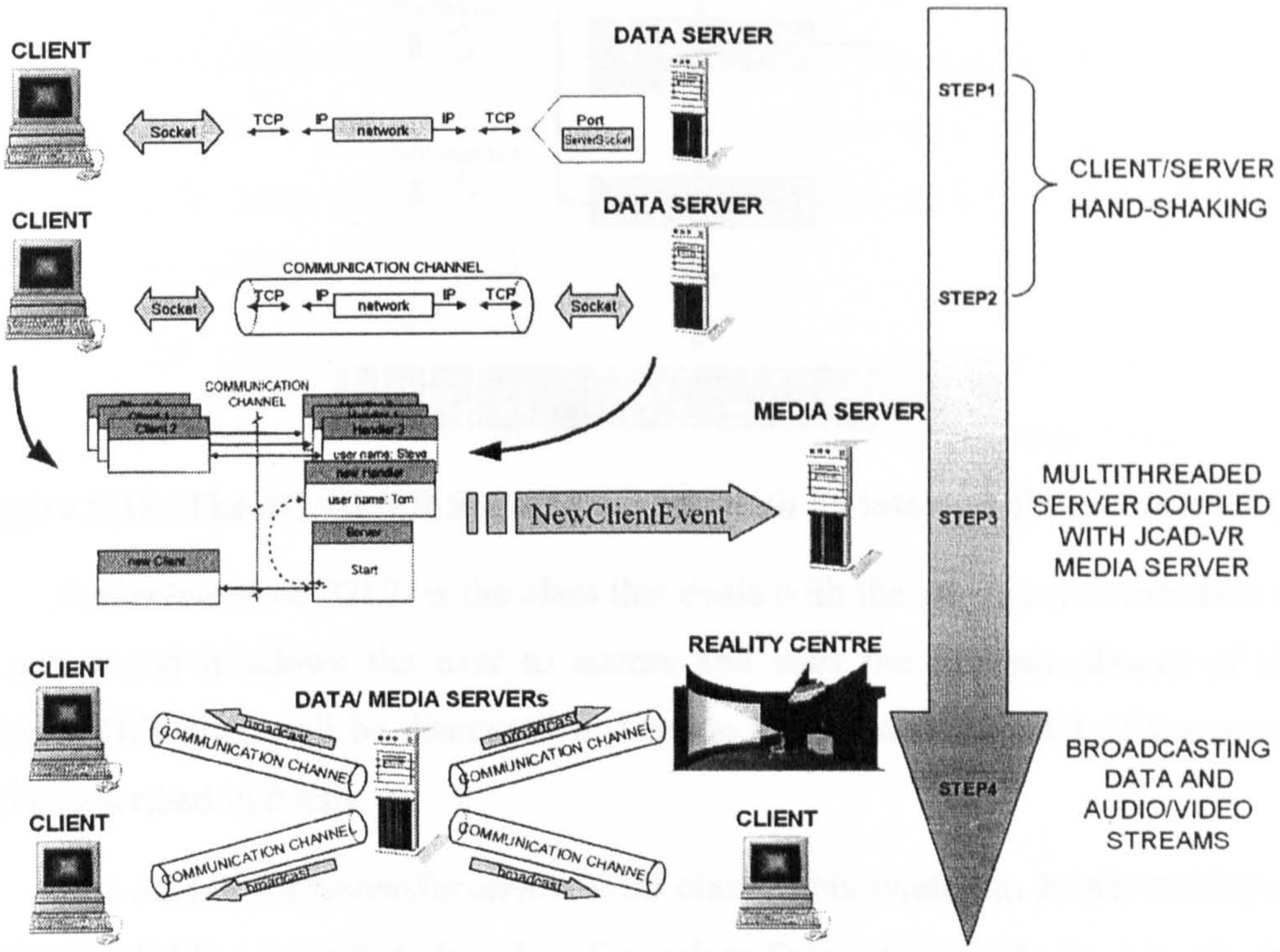


Figure 5.17: This sequence illustrates the initial phases of the communication between the clients and the server

The next sections introduce the software architecture of *JCAD-VR Server* and *JCAD-VR Media Server* describing their specific features in detail.

5.5.1 JCAD-VR Server

The *JCAD-VR Server* package is responsible for the transmission of data and geometrical information for upgrading both the Virtual Environment and the clients' collaborative tools.

It is based on the *JCAD_VRSystemServerABACUS* and the *JCAD_VRSystemHandlerABACUS* classes and on its simple Graphic User Interface (GUI). Figure 5.18 shows the internal relationship of its classes, the graphic user interface and the underlying network.

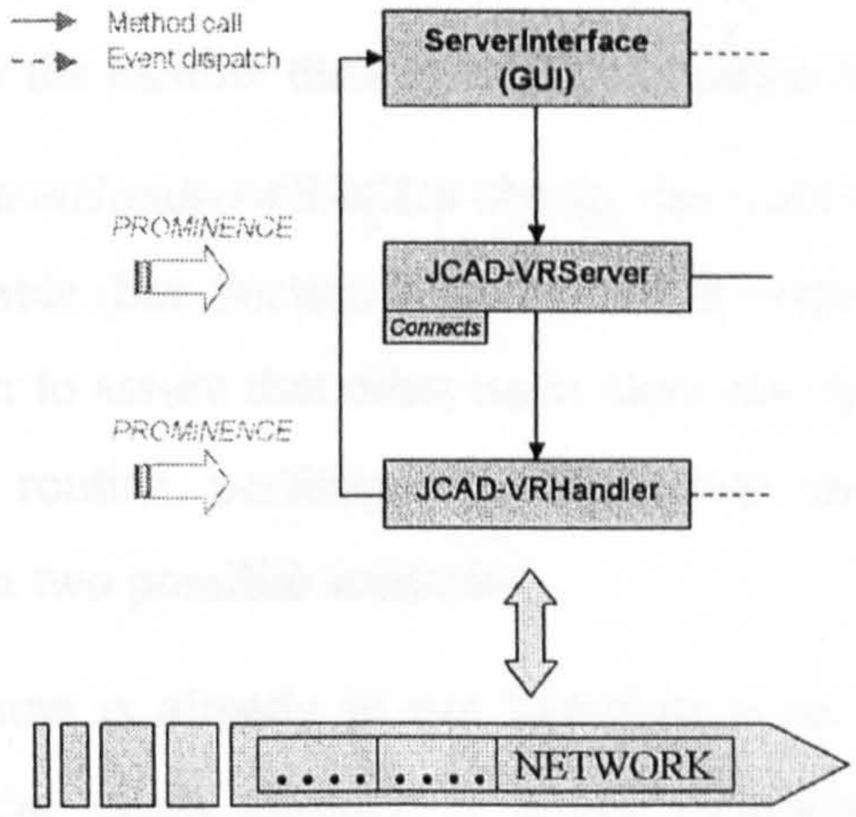


Figure 5.18: The JCAD-VR Server package with its internal classes and GUI

ServerInterface (GUI) is the class that deals with the graphic user interface of the server and it allows the user to access and start the internal classes of the package. This class will be discussed in Section 5.5.1.1 and the GUI of the server will be described in detail.

The *JCAD_VRSystemServerABACUS* class, (abbreviated to JCAD-VRServer in Figure 5.18), is called by the *ServerInterface* class and it extends the *GenericServer* of the Prominence library (See Section 5.3.1.3). This class asks for a server port through which it establishes the initial connection with the network that preparates the server for establishing communication with incoming clients. Moreover *JCAD_VRSystemServerABACUS* makes calls to the *JCAD-VR Media Server* package and to *JCAD_VRSystemHandlerABACUS*, (titled JCAD-VRHandler in Figure 5.18), passing information on the newly established clients' streams. All

the necessary operations to manage the transmission of data are then handled by the inherited *GenericServer* methods of the Prominence library.

JCAD_VRSystemHandlerABACUS is responsible for processing the messages coming from the client through the methods inherited from the *ChatSystemHandler* class (See Section 5.3.1.3), and it also extends these methods by adding new features. The internal architecture of the class, see Figure 5.19, is described as follows:

1. The class is called by *JCAD_VRSystemServerABACUS* when a new client has established the connection with the server. The client, before starting the normal transmission of data, sends their username and IP address and these are received by the handler through a *DataInputStream*.

2. *JCAD_VRSystemHandlerABACUS* checks the received username against the "routes" hashtable (See Section 5.3.1.3), which contains the list of connected clients, in order to assure that other users have not used the same login name. This checking routine, performed by the *accept* method (shown in Figure 5.19), results in two possible scenarios:

   - The username is already in use therefore *accept* sends a false boolean value to the client through a *DataOutputStream* notifying that the connection has been refused.

   - The checking has been successful, the client's username is included in "routes" and the inherited methods of the *ChatSystemHandler* class (See Section 5.3.1.3) prepare their components to process the messages coming from the client.

3. Once the client has been accepted, the *NexthandlerNum* method defines a name for the new thread that will be responsible for handling all the messages coming from the new client as an independent process,. This method performs a routine that attaches a number from 1 to 99, to the new thread. This number is also the client's identification number, ID, and it is sent back to the client, together with the notification of acceptance, through a *DataOutputStream*. The *NexthandlerNum* method also sends an event, *ServerUserEvent*, to *JCAD-VR MediaServer* with information on the status of

the client, for example if they are connected or disconnected, their username, IP address and ID number.

4. Both username and IP address are finally sent to the *ServerInterface* class that manages the GUI of the server and holds the lists of clients connected together with their IP addresses.

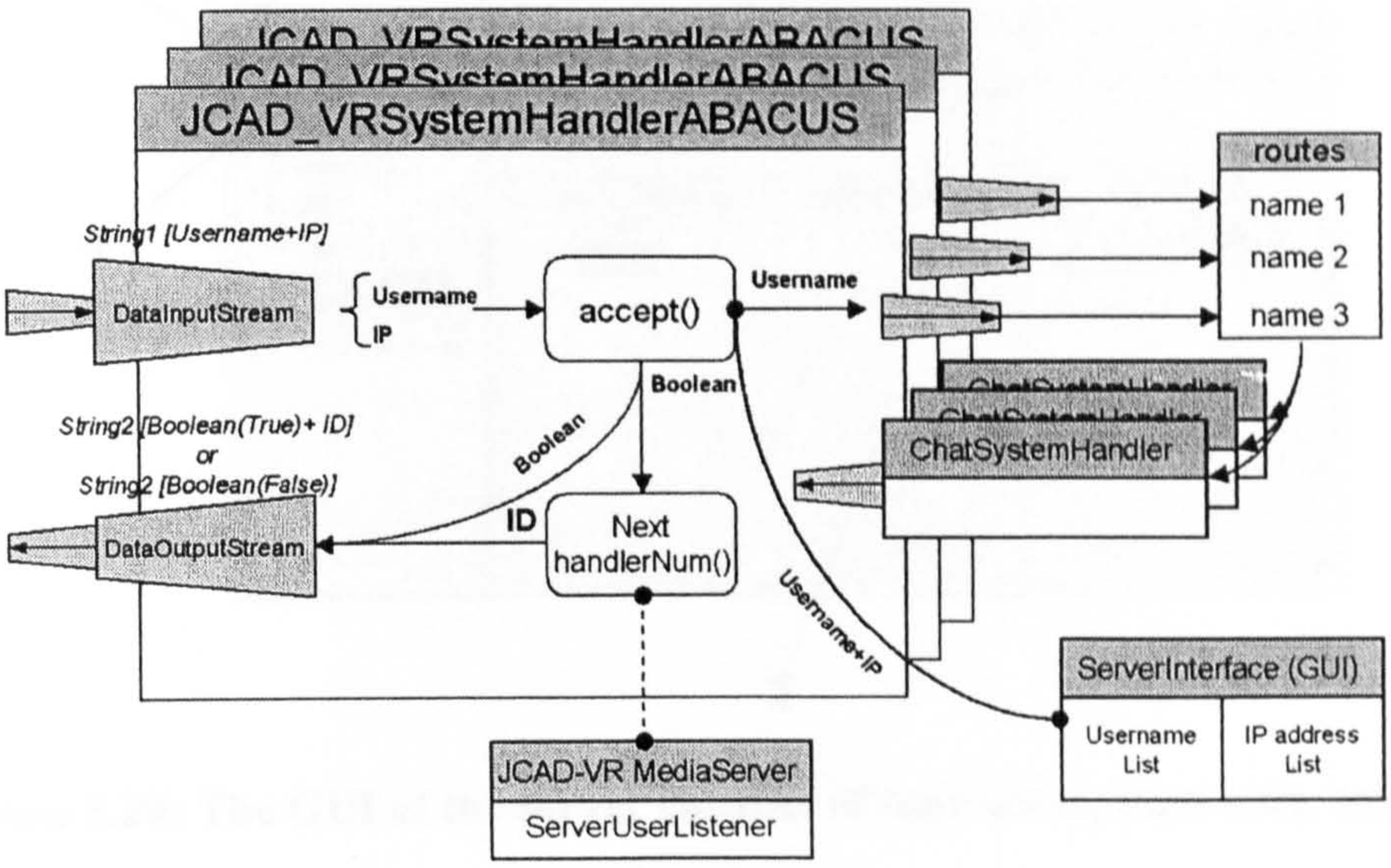


Figure 5.19: The internal architecture of JCAD_VRSystemHandlerABACUS

As in *GenericServer* and *ChatSystemHandler* classes (See Section 5.3.1.3), the *JCAD-VR Server* is not aware of the data content received from the client since the streams are managed at a high level. The type of data that is transmitted and its composition will be illustrated in detail in Chapter 6 with a description of the client side of the network package of JCAD-VR.

The following section will complete the description of the *JCAD-VR Server* package illustrating its GUI.

5.5.1.1 GUI

The *ServerInterface* class extends the *JFrame* class that is part of the standard *Swing* package of Java™. *Swing* offers a number of customisable classes to help implement the GUI of a Java™ application including the *JFrame* class, which is used as container of GUI elements, i.e. buttons, tables, menus etc. (Eckel, 2000).

The *ServerInterface* class is responsible for commanding important functions of the server including the initialisation of its components and the opening of the connection with the network. Visually its window is divided into four main areas (See Figure 5.20):

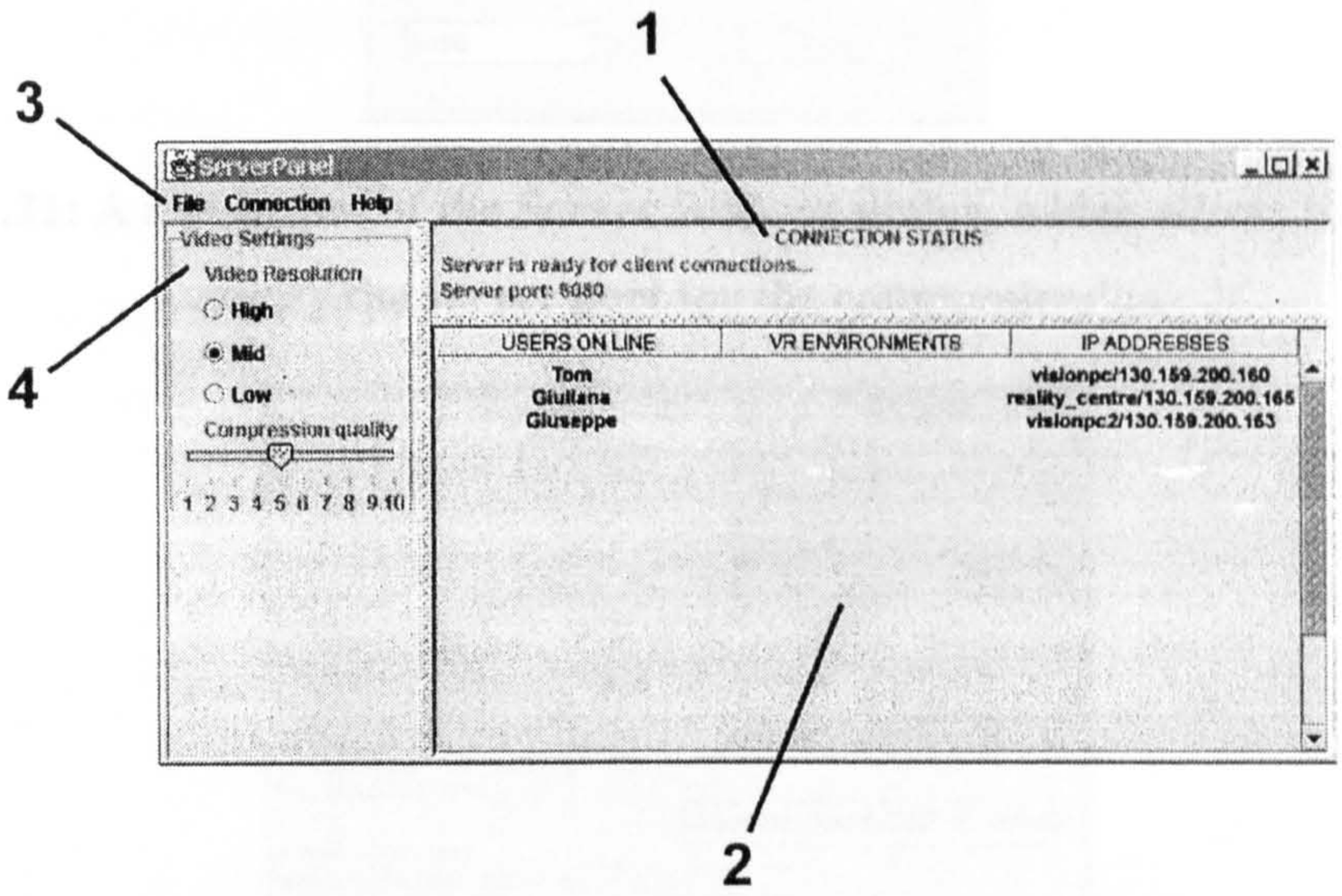


Figure 5.20: The GUI of the server consists of four areas, each with different functions

1. Area 1 allows the user to check the connection status of the server and it shows the server port currently used to communicate with the network.
2. Area 2 contains a table that holds lists of connected clients and their IP addresses.
3. The menu bar includes the three following choices:
    a. **File**: it accesses the *Exit* dialog that allows the user to close the server and to quit its GUI.
    b. **Connection**: it contains three options, *Start Server* to start the communication, *Close Server* to close the communication without exiting the GUI and *Choose port* that has a default value of 8080. This allows the user to specify another server port through a server settings dialog (See Figure 5.21).
    c. **Help**: it provides information about the server package and the contact details of the JCAD-VR authors (See Figure 5.22).

4. In area 4 are the video settings for the *JCAD-VR Media Server* package. A radio button and a slider allow the user to set the resolution and the compression quality of the video streamed by the server.

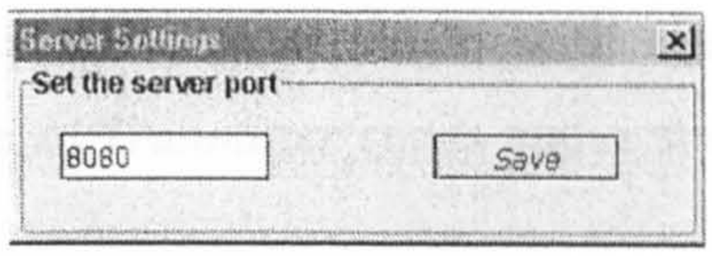


Figure 5.21: A screenshot of the Server Settings dialog, which allows the user to specify the server port for the communication

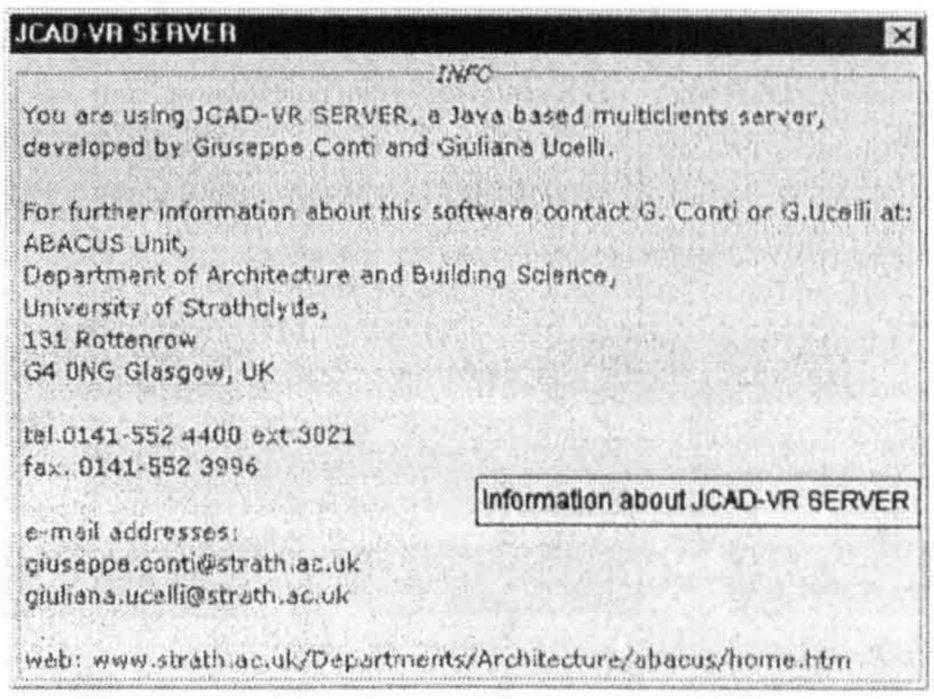


Figure 5.22: A screenshot of the Info panel that provides general information about the JCAD-VR Server package and the contact details of the authors

5.5.2 JCAD-VR Media Server

The *JCAD-VR Media Server* package is responsible for broadcasting audio and video streams for the video conferencing tool of JCAD-VR.

It is based on two types of classes: the first type handles the transmission of RTP streams (See Section 5.3.2.1) and includes the *AVReceiver* and *AVTransmitter* classes, while the second type includes "service" classes such as *MediaServer* and *Broadcaster*, which manage and organise the transfer of the streams from a particular client to the correct recipients. In other words the former classes allow the input and output of data streams and the setting up of RTP connections, while the latter administer the data to the right clients.

AVReceiver and *AVTranmitter* are customised versions of classes developed by Sun Microsystems, Inc. to make use of video conferencing tools easier.

Implementing RTP sessions with several distributed clients and streams can be a tedious and complicated task but JMF™ (See Section 5.3.2) provides readily available and easily extendible solutions to the management of video conferencing applications. In particular *AVReceiver* deals with the reception of incoming audio/video streams, while *AVTranmitter* sends audio and video streams out to the underlying network.

Figure 5.23 shows the whole server architecture of JCAD-VR emphasising the relationships between the classes of the *JCAD-VR Server* package, on the left of the figure, and of the *JCAD-VR Media Server*, on the right.

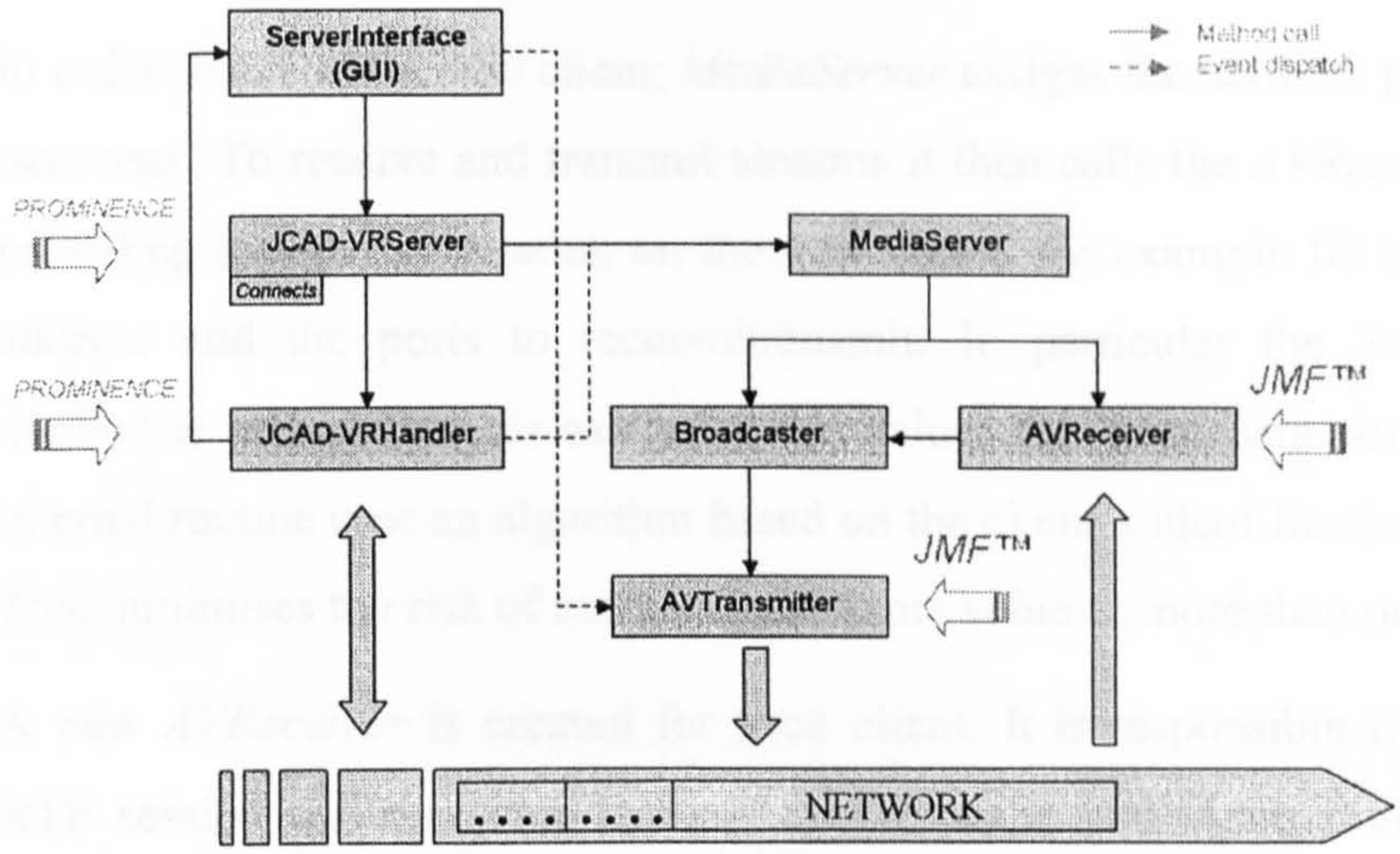


Figure 5.23: The server architecture with the JCAD-VR Server package, (left), and JCAD-VR Media Server package, (right)

The *JCAD_VRSystemServerABACUS* class, (titled JCAD-VRServer in Figure 5.23), is part of the *JCAD-VR Server* package (See Section 5.5.1), and is responsible for calling the *MediaServer* class of the *JCAD-VR Media Server* package once a new client has asked for connection to the server. Then the *JCAD_VRSystemHandlerABACUS*, (titled JCAD-VRHandler in Figure 5.23), performs the username check, establishes the network connection with the new client and finally sends an event, (titled *ServerUserEvent*). When this arrives at the *MediaServer* it makes it start the procedures to receive and transmit audio/video streams from/to the clients. As already mentioned in Section 5.5.1.1, the *ServerInterface* contains the GUI video settings for the *JCAD-VR Media Server*

package. Specifically a radio button permits the user to set the resolution of the video, while a slider regulates its compression quality. The *AVTransmitter* class of the *JCAD-VR Media Server* then sends out these video streams.

The internal architecture of the *JCAD-VR Media Server* package is illustrated in further detail in Figure 5.24. The description of its internal classes will follow this logical progression:

1. The *MediaServer* class listens for notification of connections or disconnections of users through the *ServerUserEvent* sent by the *JCAD-VR Server* package.

2. In order to connect a new client, *MediaServer* assigns the network ports to the new user. To receive and transmit streams it then calls the *AVReceiver* class providing it with information on the new client, for example ID number, IP address and the ports to receive/transmit. In particular the *ServerPortR* method is responsible for assigning the values for these data ports, and its internal routine uses an algorithm based on the client's identification number. This minimises the risk of assigning the same value to more than one client.

3. A new *AVReceiver* is created for each client. It is responsible for opening RTP session and receiving streams of live audio and video from the new client RTP. These streams, (referred to as *data source* (*ds*) in Figure 5.24), together with other information about the new client, are sent to the *Broadcaster* class. *Broadcaster* creates a new handler for the new client that will hold all the information on that user. All the *Broadcaster* handlers, holding information and data sources of all the clients currently connected, are then stored in the "UserList" of *MediaServer*. "UserList" is a container of *Broadcaster* objects and it can be accessed even from outside the *MediaServer* class.

4. After the creation of a new handler of *Broadcaster* for the client, the method *addListUsers* of *AVReceiver* upgrades the "UserList" of *MediaServer*. In particular *addListUsers* performs three actions on "UserList":

    a. It notifies to the other clients that a new user has to receive their audio/video streams.

b. It retrieves information about the clients already connected in order to include them as recipients of the new user's streams.

c. It includes the *Broadcaster* handler of the new client in the list.

5. Finally the *AVTransmitter* class is called to stream to the network the data source of the new client through RTP streams. Its *addTarget* method, filled by the "UserList", is responsible for holding an internal list of all the possible clients ready to receive these RTP streams and it also calls other internal methods that transmit the audio/video streams to these clients.

6. *MediaServer* is also responsible for the disconnection procedures. When the *ServerUserEvent* notifies that a client has left, the "UserList" is upgraded deleting the appropriate handler of the *Broadcaster* class and the relative *AVReceiver* and *AVTransmitter* classes are quit.

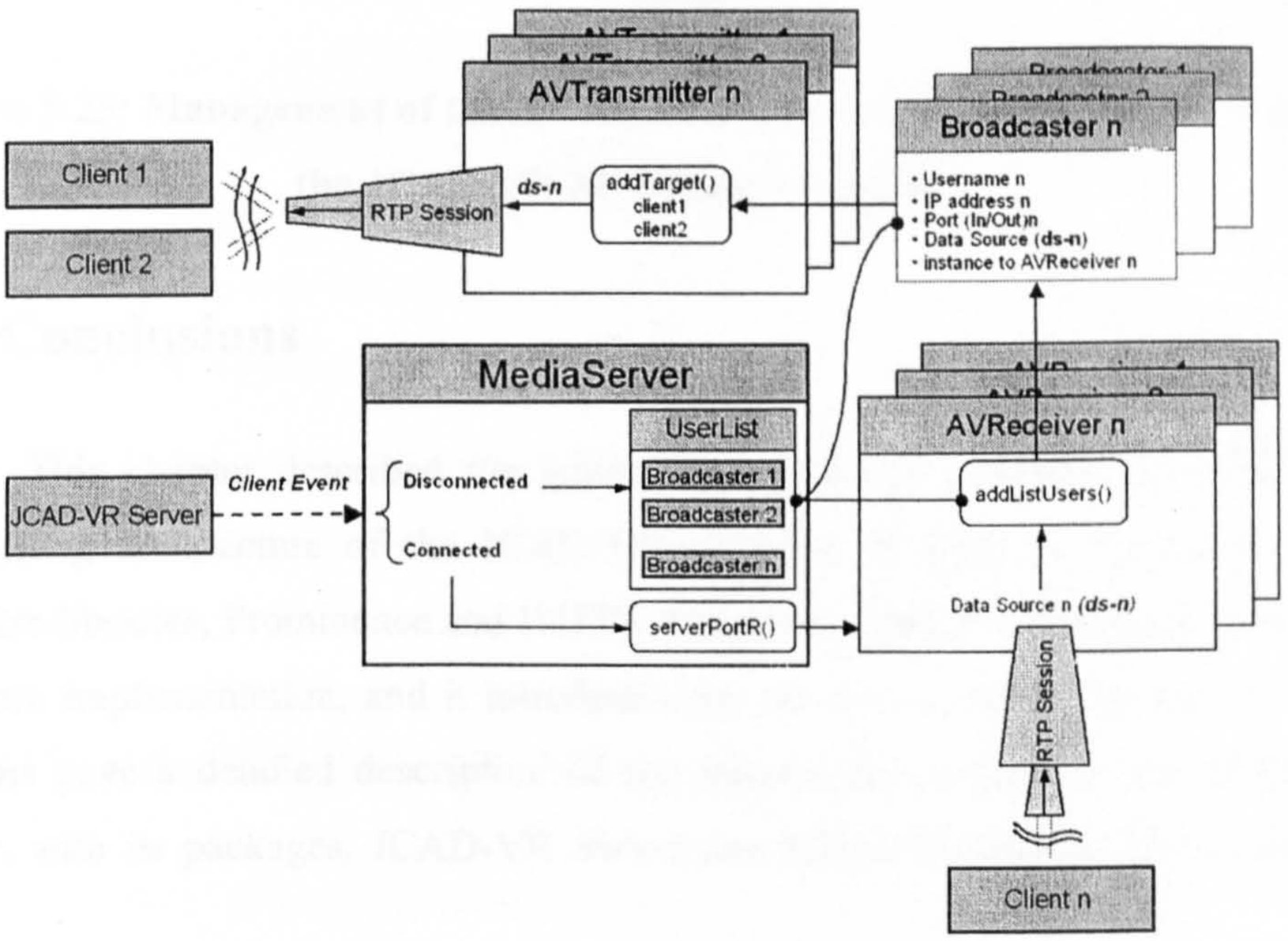


Figure 5.24: The internal architecture of the JCAD-VR Media Server package

In conclusion, as shown in Figure 5.25, the *JCAD-VR Media Server* package receives audio/video streams from the client N through the handler N of the *AVReceiver* class and it sends audio/video coming from the same client N to the other clients through the handler N of the *AVTransmitter* class.

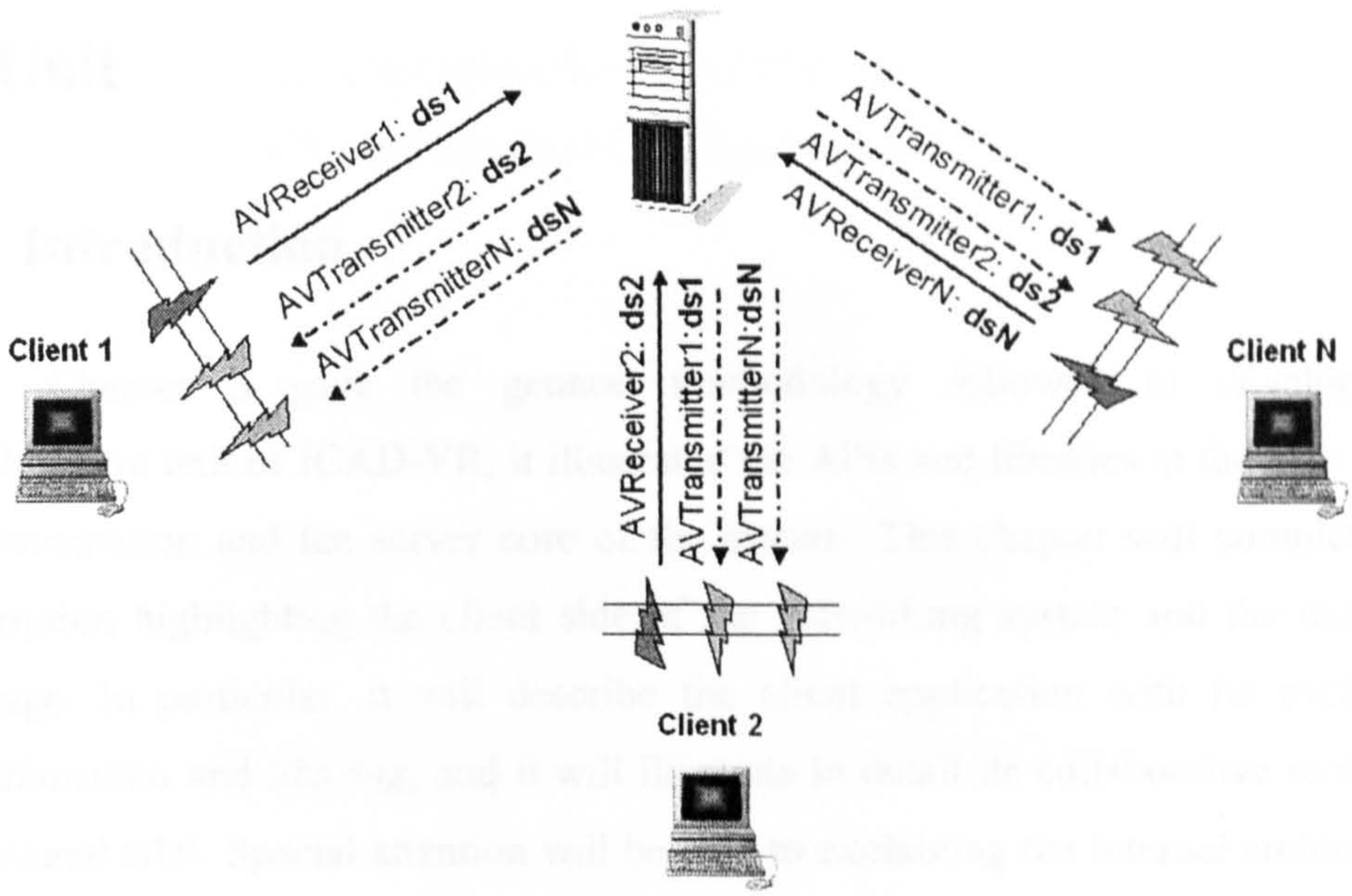


Figure 5.25: Management of the AVReceiver and the AVTransmitter classes in the JCAD-VR Media Server package

5.6 Conclusions

This chapter described the general methodology followed to develop the networking architecture of the JCAD-VR software. It gave an overview of the software libraries, Prominence and JMF™, which were used as starting points for the software implementation, and it introduced the server core of the system. The last sections gave a detailed description of the internal architecture of the JCAD-VR server, with its packages, JCAD-VR Server and JCAD-VR Media Server, and the GUI.

Chapter 6 will present the client-side of the networked application of JCAD-VR with its collaborative tools and GUI, and it will illustrate the final package, called Database Core, which handles the internal database of objects created and modified in the VE.

6 The Implementation of the Collaborative Client Unit

6.1 Introduction

Chapter 5 gave the general methodology followed to develop the collaborative unit of JCAD-VR, it illustrated the APIs and libraries at the base of its implementation and the server core of the system. This chapter will complete the description highlighting the client side of the networking system and the database package. In particular, it will describe the client application with its packages, *Collaboration* and *Sharing*, and it will illustrate in detail its collaborative tools and the related GUI. Special attention will be paid to explaining the internal architecture of the collaborative tools and then showing the relationship between the network-based system and the Virtual Environment. The description of the collaborative unit will be completed with information on the client's video conferencing core and on how network persistence has been resolved in JCAD-VR. As in the previous chapter, several diagrams will augment the explanation of the internal classes of the packages.

Finally, a section will explain the database core of JCAD-VR which handles the internal database of objects in the virtual scene and manages the saving and loading routines.

6.2 The Implementation of the Client Unit

The client side of the networked unit of JCAD-VR is based on the *Network*, *Sharing* and *Data* cores. The general framework shows these packages and their functions schematically in Figure 5.16. The *Network* core handles the *Collaboration Module* which includes the collaborative tools of the system and the video conferencing package. The *Sharing* core is responsible for the streaming of geometrical information related to the virtual world and it regulates the priority of the users interacting with the virtual objects. In addition the system has been augmented

with a mechanism that assures objects persistence even when the program is not running.

The client side of the networked JCAD-VR system has been implemented by using the APIs and libraries described in Chapter 5 (See Section 5.3).

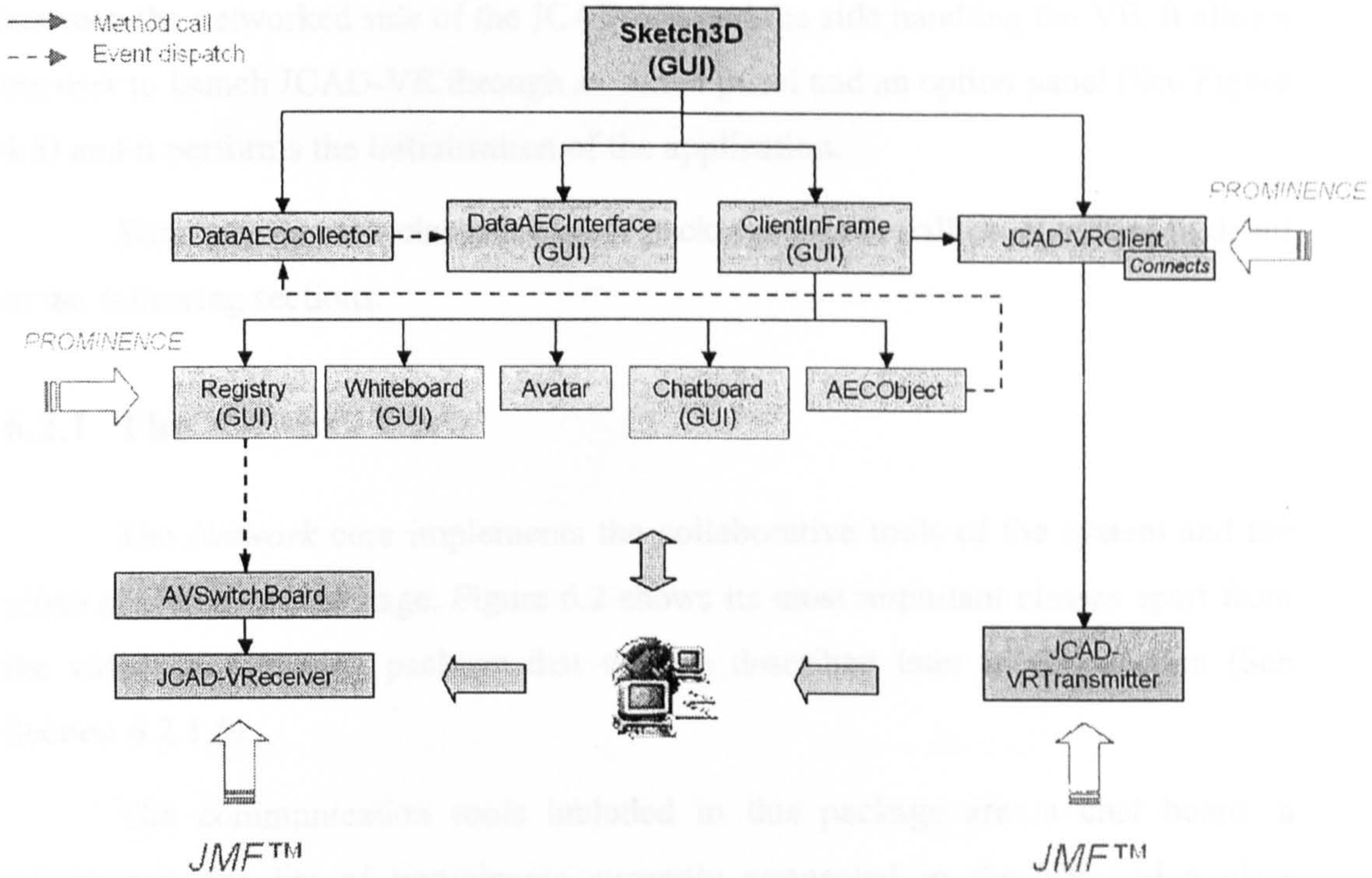


Figure 6.1: The most important classes of the client side of the JCAD-VR networked unit

Some of the collaborative tools of the client application are represented visually within the virtual scene but, as already mentioned in Chapter 1 (See Section 1.4), the management, implementation, visualisation, creation and manipulation of objects in the virtual world are outside the scope of this thesis. Nevertheless, when necessary, some screenshots of the VR system will be provided to help to explain and visualise the results of network communication. The implementation of the VE is described in detail in the companion thesis that runs parallel with this project (Conti, 2002).

The client application includes the *Data Module* that manages the database internal to every user's computer, it allows input from CAD/CAAD packages and it performs loading and saving routines from/to a proprietary file format.

Figure 6.1 illustrates the internal architecture of the packages and their mutual relationships. *Sketch3D* is the initial class of the software and it is shared between the networked side of the JCAD-VR and the side handling the VE. It allows the user to launch JCAD-VR through an initial panel and an option panel (See Figure 4.5) and it performs the initialisation of the application.

Sketch3D up calls classes of other packages which will be described in detail in the following sections.

6.2.1 The Network Core

The *Network* core implements the collaborative tools of the system and the video conferencing package. Figure 6.2 shows its most important classes apart from the video conferencing package that will be described later in this section (See Section 6.2.1.5).

The communication tools included in this package are: a chat board, a whiteboard, the list of participants currently connected in the VE and a class handling the user's avatar. *ChatboardABACUS*, *WhiteboardABACUS*, *RegistryABACUS* and *AvatarsBehaviour* classes, (which in the figure are shortened Chatboard, Whiteboard, Registry and Avatar respectively), are the implementation of these tools and they will be illustrated in detail in the following sections (See Sections 6.2.1.1, 6.2.1.2, 6.2.1.3 and 6.2.1.4).

The *Network* package implements the client side of the server/client architecture in the JCAD-VR system and it is based on the networking framework included in the Prominence library (See Section 5.3.1).

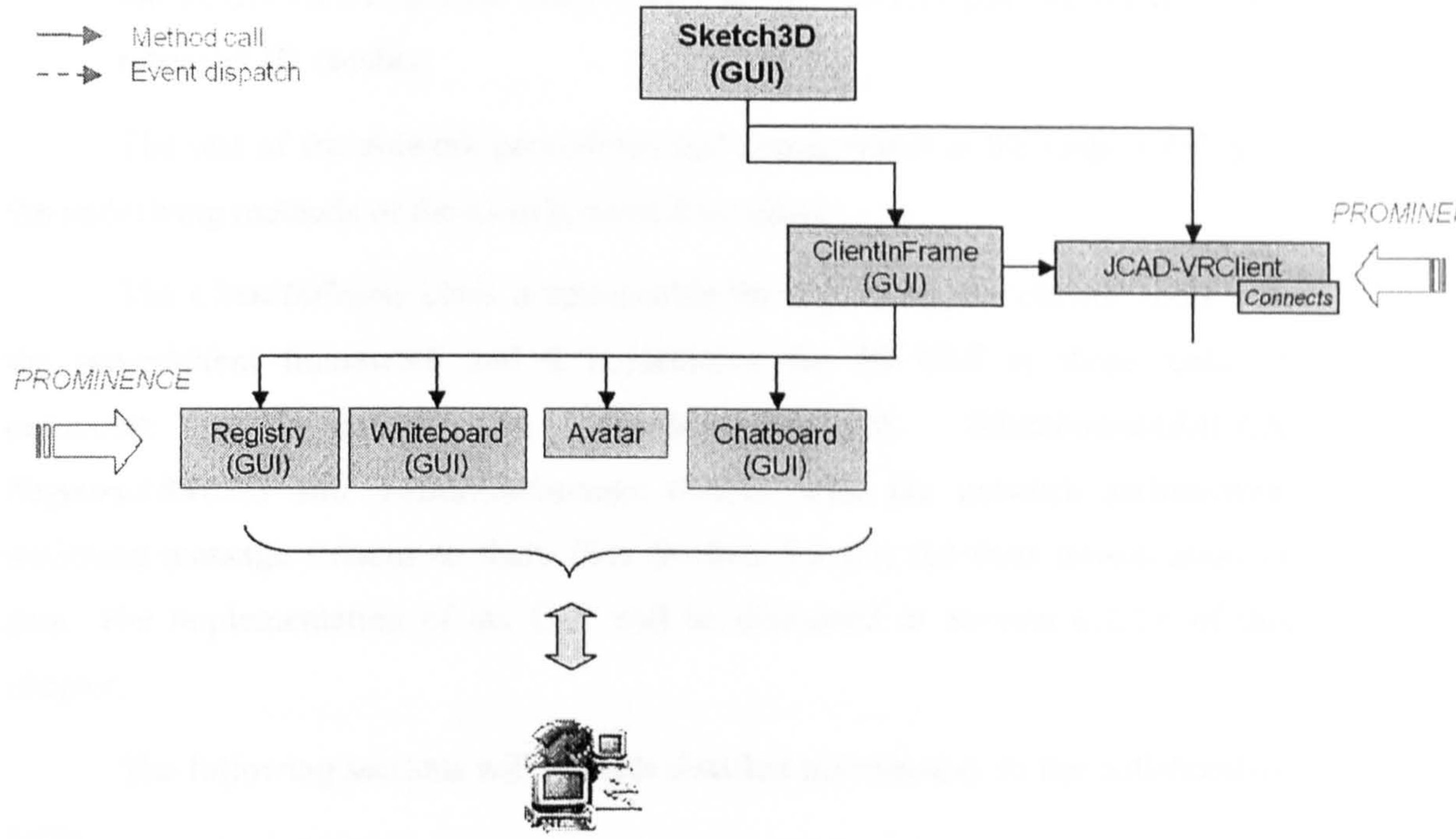


Figure 6.2: The most important classes of the Network core apart from the video conferencing package

The system description will begin with the classes *Sketch3D* that calls both the *JCAD-VRSystemClient* class, (called JCAD-VRClient in Figure 6.2), and the *ClientInFrame* class. The *JCAD-VRSystemClient* class extends the *ChatSystemClient* (See Section 5.3.1.3) class and it handles all the routines necessary to set up the network sockets and establishes the connection with the server. In particular, this class allows the connection with the *JCAD-VR server* (See Section 5.5.1) and it attempts to register the client name specified in the initial panel of JCAD-VR with the server (See Figure 4.5). It then waits for the notification of acceptance/refusal from the *JCAD-VR server*. *JCAD-VRSystemClient* responds to the notice of the server as follows:

a. If the server has refused its registration it closes the connection and quits the application.

b. If the server has accepted its registration, after sending its ID number, it checks whether the user made the video conferencing tool active. It then calls

the *JCAD-VRTransmitter* class (See Section 6.2.1.5) passing on its newly retrieved ID number.

The rest of the network procedures and management is the responsibility of the underlying methods of the *ChatSystemClient* class.

The *ClientInFrame* class is responsible for registering the clients' tools with the server/client framework and it implements the 2D GUI of these tools. In particular, it registers the *ChatboardABACUS*, *WhiteboardABACUS*, *RegistryABACUS* and *AvatarsBehaviour* classes with the network architecture, assigning message streams to them (See Section 5.3.1.1) for their transmission of data. The implementation of the GUI will be discussed in Section 6.2.1.6 of this chapter.

The following sections will provide detailed information on the collaborative tools.

6.2.1.1 Chatboard

The *ChatboardABACUS* class is a simple chat board that customises the implementation of the Chatboard class of Hughes et al., (1997). It implements the *ChatClient* interface of the Prominence library (See Section 5.3.1.3) thus it is included in the general communication framework. It uses message streams to transmit its data through the communication channel. It communicates with the currently connected remote clients *ChatboardABACUS* classes providing a quick and effective means of communication.

The communication channel transmits *Strings*, or a sequence of characters, carrying the message typed by the user, and in order to make communication among the participants easier, it also attaches the name of the client that sent the message.

The incoming messages can be visualised on either a traditional 2D panel (See Section 6.2.1.6) or as 3D text within the Virtual Environment. Using the 2D interface messages can be written in the text area provided or alternatively they can be simply typed by clicking on the 3D chat icon in the virtual scene. 3D chat is managed by the *VirtualMenu_Chat* class (See Section 6.5.3.3 of the companion thesis Conti, 2002).

6.2.1.2 Whiteboard

Based on the *Whiteboard* class of Hughes et al., (1997), the *WhiteboardABACUS* class provides a surface on which the user can sketch freely using a normal mouse. The principle behind this tool is that the scribbles are generated by sequences of points whose x and y coordinates can be tracked and then sent, each time a user releases the appropriate button of the mouse.

WhiteboardABACUS implements the *ChatClient* interface (See Section 5.3.1.3) and the underlying network architecture allows the remote *WhiteboardABACUS* classes of the other participants to receive the coordinates of the points drawn by the other users in real-time.

WhiteboardABACUS allows drawing using different colours which are retrievable from a colour palette. The use of colours has been recognised by the author to be an advantage for the clarity of the communication since it helps to differentiate between each user's sketches on the common drawing surface.

The GUI of the class is very simple and intuitive. It features only a black board which is sensitive to mouse movements, a button for accessing the colour palette and a delete button which can erase the content of the whole black board surface on demand (See Section 6.2.1.6).

6.2.1.3 Registry

The *RegistryABACUS* class is a tool that allows each client to determine which users are currently connected to the session. It is a customised version of the *Registry* class developed by Hughes et al., (1997) and along with the other tools it implements the *ChatClient* interface (See Section 5.3.1.3) to access message streams (See Section 5.3.1.1) for communication. *RegistryABACUS* has a particular function which is to listen for announcements from the server that users are joining or leaving the collaborative session and it shows the participants' names visually in a 2D panel (See Section 6.2.1.6).

RegistryABACUS class is then responsible for notifying changes to the current list of users and so it fires an event, called *UserEvent.* This carries information on the user's status, whether they are joining or leaving, their username

and ID number, for the video conferencing package (See Section 6.2.1.5) and the *AvatarsBehaviour* class (See Section 6.2.1.4).

6.2.1.4 Avatar

The *AvatarsBehaviour* class handles the dataset necessary to run the avatars mechanism. First it creates a 3D avatar in the VE and then it reads the position of the point of view of the user in the virtual scene and it sends it to the network module. When it receives the position of the other users from the communication channel, it upgrades the relevant values in the Virtual World.

This class implements the *ChatClient* (See Section 5.3.1.3) and the *UserListener* interfaces and so it is therefore called on every time a user in the virtual world changes their position or logs in or out. The *UserListener* interface provides the means for listening events of the *UserEvent* type which are fired by the *RegistryABACUS* class (See Section 6.2.1.3) and then the relevant user's avatar is created or deleted from the VE accordingly. Through the *UserEvent* method titled *getUserName* it is possible to retrieve the username that then will be shown in the Virtual World next to the avatar to easily identify the participants.

To optimise the performance of the avatars mechanism, its position coordinates are received from the network and are queued and processed every time the *processStimulus* method is called on. This makes sure that the avatars' positions are not upgraded more frequently than every 30 milliseconds, freeing the use of the network for other communication.

Finally the *Avatar* class manages the user's embodiment in the Virtual World, handles the avatar's visual components and the upgrading of the virtual scene (See Section 7.7 of the companion thesis Conti, 2002). Figure 6.7 in Section 6.2.1.6 shows a screenshot of JCAD-VR depicting an avatar of human appearance with a 3D text area which shows the log in name of the user.

6.2.1.5 Video Conferencing Package

The video conferencing package is responsible for sending audio and video streams to the *JCAD-VR Media Server* (See Section 5.5.2). Figure 6.3 shows the internal classes of the package with the *JCAD-VReceiver* and *JCAD-VRTransmitter*

classes which are responsible for handling the reception and transmission of RTP streams (See Section 5.3.2.1). The *AVSwitchBoard* class that manages the handlers to the *JCAD-VReceiver* class and links them to the rest of the networked system is also shown. Figure 6.4 illustrates in greater detail the internal architecture of the package dividing the reception from the transmission procedure.

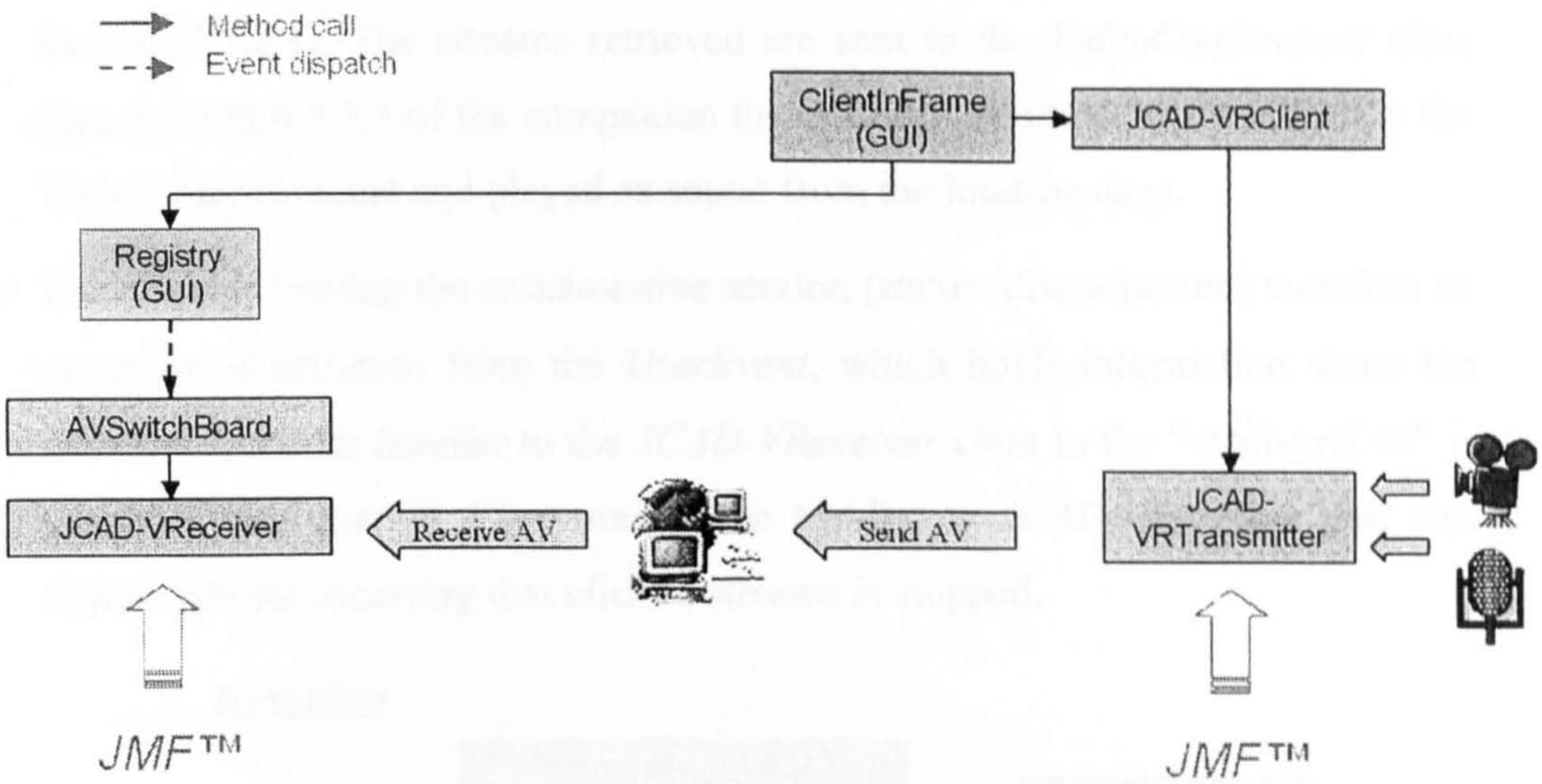


Figure 6.3: The most important internal classes of the video conferencing package

As described, starting from the reception process the *RegistryABACUS* class (See Section 6.2.1.3) sends a new *UserEvent* to announce that a new client has joined the collaborative session. The *AVSwitchBoard* class, which implements *UserListener*, listens for this announcement and acquires information on the new participant from the *UserEvent* only if the video conferencing facility has been activated by the user in the initial option panel (See Figure 4.5). Then, according to the "status" of the client, one of the following two scenarios will take place:

1. The client has just joined the session (status=connected). The *clientPortToReceive* method of the *AVSwitchBoard* class, (shortened in Figure 6.4 to *clientPortR*), creates a port number for the reception of incoming streams using an algorithm based on the ID number of the new client. Then *AVSwitchBoard* calls the *JCAD-VReceiver* class and creates a new handler of this class, passing the port for the opening of the RTP session. Finally the handler to

JCAD-VReceiver class is stored in a list, (called "receiversList" in Figure 6.4), in order to ease its retrieval in case of the disconnection of the relevant user. The *JCAD-VReceiver* class is the equivalent for the client of the *AVReceiver* class included in the server package (See Section 5.5.2). It is responsible for listening out for RTP streams sent by the *JCAD-VR Media Server* and it performs all the connections necessary with the underlying network using the RTP protocol (See Section 5.3.2.1). The streams retrieved are sent to the *VideoConferencer* class (See Section 6.5.3.3 of the companion thesis Conti, 2002) to be visualised in the Virtual Environment and played as sound from the loudspeakers.

2. The client is leaving the collaborative session (status=disconnected) therefore its username is retrieved from the *UserEvent*, which holds information about the relevant client. Its handler to the *JCAD-VReceiver* class in the "receiversList" is identified and deleted. Concurrently the handler to *JCAD-VReceiver* that was responsible for receiving that client's streams is stopped.

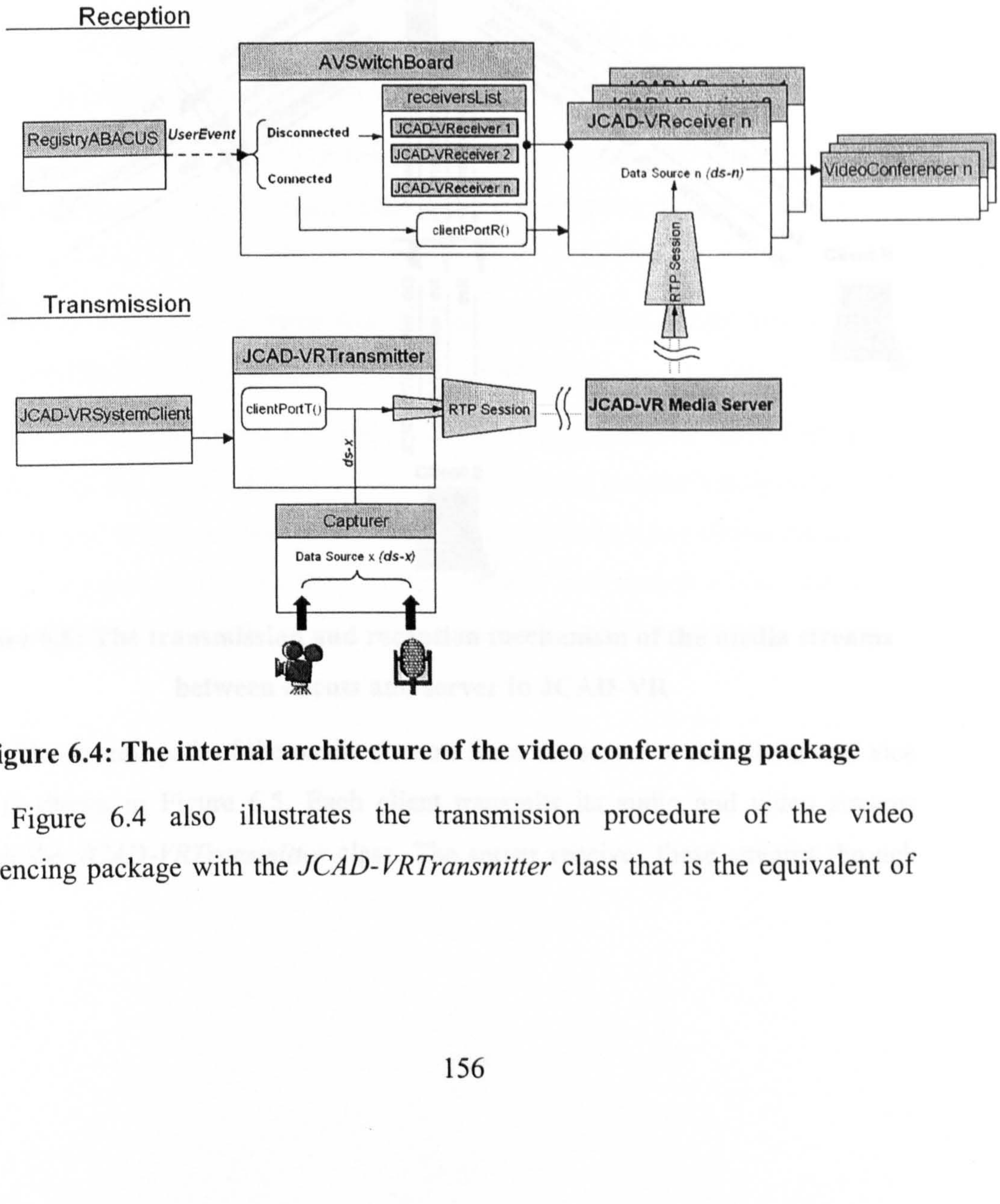


Figure 6.4: The internal architecture of the video conferencing package

Figure 6.4 also illustrates the transmission procedure of the video conferencing package with the *JCAD-VRTransmitter* class that is the equivalent of

the *AVTransmitter* class for the client (See Section 5.5.2). It is responsible for sending audio and video streams through the RTP protocol (See Section 5.3.2.1).

The *JCAD-VRSystemClient* class (See Section 6.2.1) activates the transmission of media streams after establishing the connection with the server and it then calls the *JCAD-VRTransmitter* class and passes the user's ID number to it. The *clientPortToTransmit* method of *JCAD-VRTransmitter*, (shortened *clientPortT* in Figure 6.4), is then called to assign a port for the transmission to the server. Concurrently a call to the *Capturer* class (See Section 6.5.3.3 of the companion thesis Conti, 2002) is performed in order to start capturing the audio and video streams from the digital camera and microphone connected to the computer. These media streams are eventually sent to the *JCAD-VR Media Server* through a RTP session.

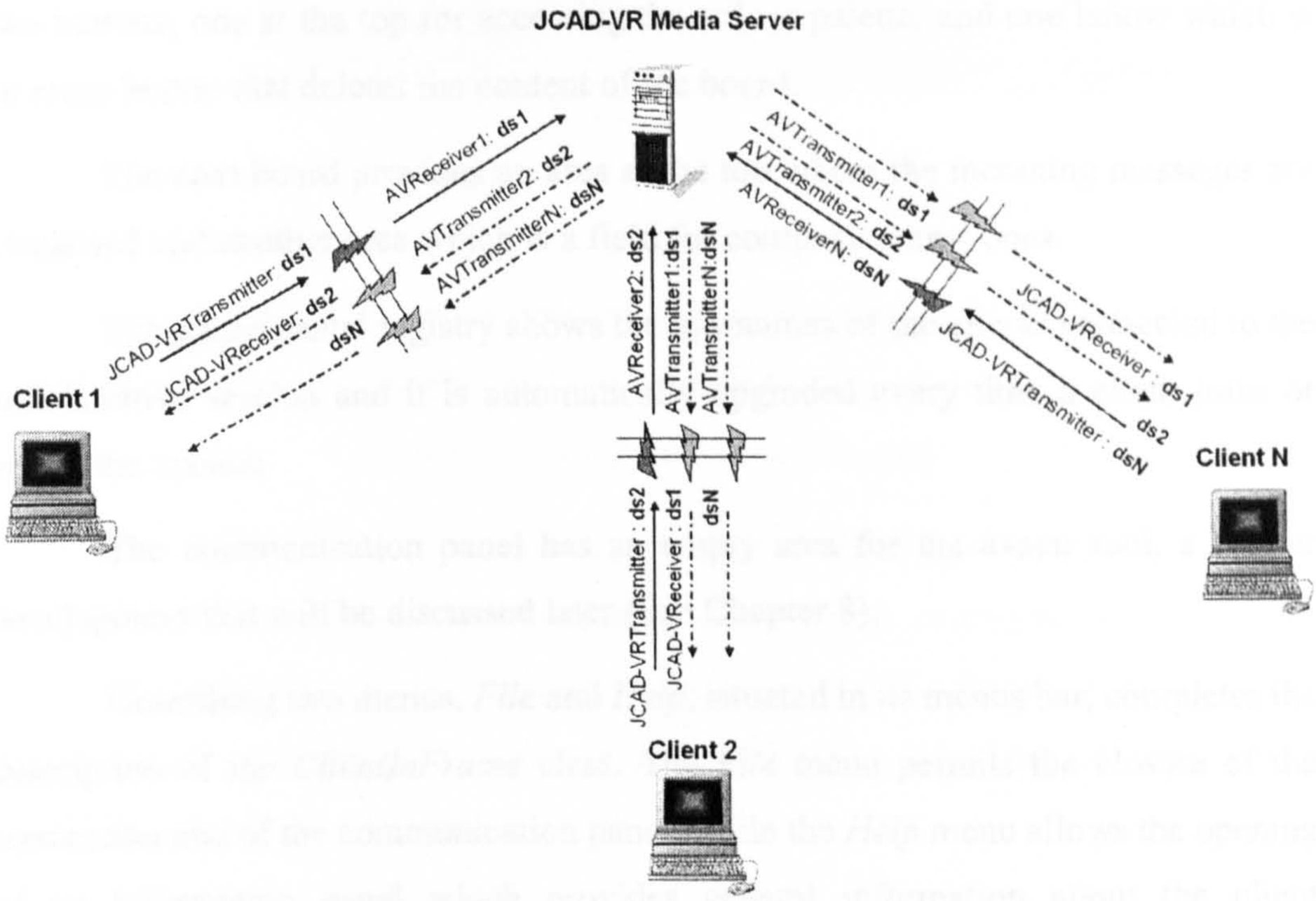


Figure 6.5: The transmission and reception mechanism of the media streams between clients and server in JCAD-VR

The overall path of the media streams from the server to the clients and vice versa is shown in Figure 6.5. Each client transmits its audio and video streams through the *JCAD-VRTransmitter* class. The server receives these streams through

the *AVReceiver* class dedicated to this client (See Section 5.5.2) while an *AVTransmitter* per client accomplishes the transmission of the datasources originated by the remote participants. The initial client eventually receives audio and video streams from the other users through the handlers of the *JCAD-VReceiver* class which are dedicated to each remote client.

6.2.1.6 GUI

As already mentioned the *ClientInFrame* class (See Section 6.2.1) manages the GUI of the *ChatboardABACUS*, *WhiteboardABACUS* and *RegistryABACUS* classes (See Sections 6.2.1.1, 6.2.1.2 and 6.2.1.3). It extends the *JFrame* class, which is a standard class of Java™ (Eckel, 2000), and it implements a multi function panel. This panel (shown in Figure 6.6) allows the user to interact with the 2D sketch board, the chat board and to monitor the list of participants. The sketch board also provides two buttons, one at the top for accessing the colour palette, and one below which is an erase button that deletes the content of the board.

The chat board provides an area at the top where the incoming messages are visualised and another area which is a field for composing new ones.

The participants' registry shows the usernames of the clients connected to the collaborative session and it is automatically upgraded every time a client joins or leaves the session.

The communication panel has an empty area for the avatar tool, a further development that will be discussed later (See Chapter 8).

Describing two menus, *File* and *Help*, situated in its menus bar, completes the description of the *ClientInFrame* class. The *File* menu permits the closure of the connection and of the communication panel, while the *Help* menu allows the opening of an Information panel which provides general information about the client application of the JCAD-VR software and contact details of the authors.

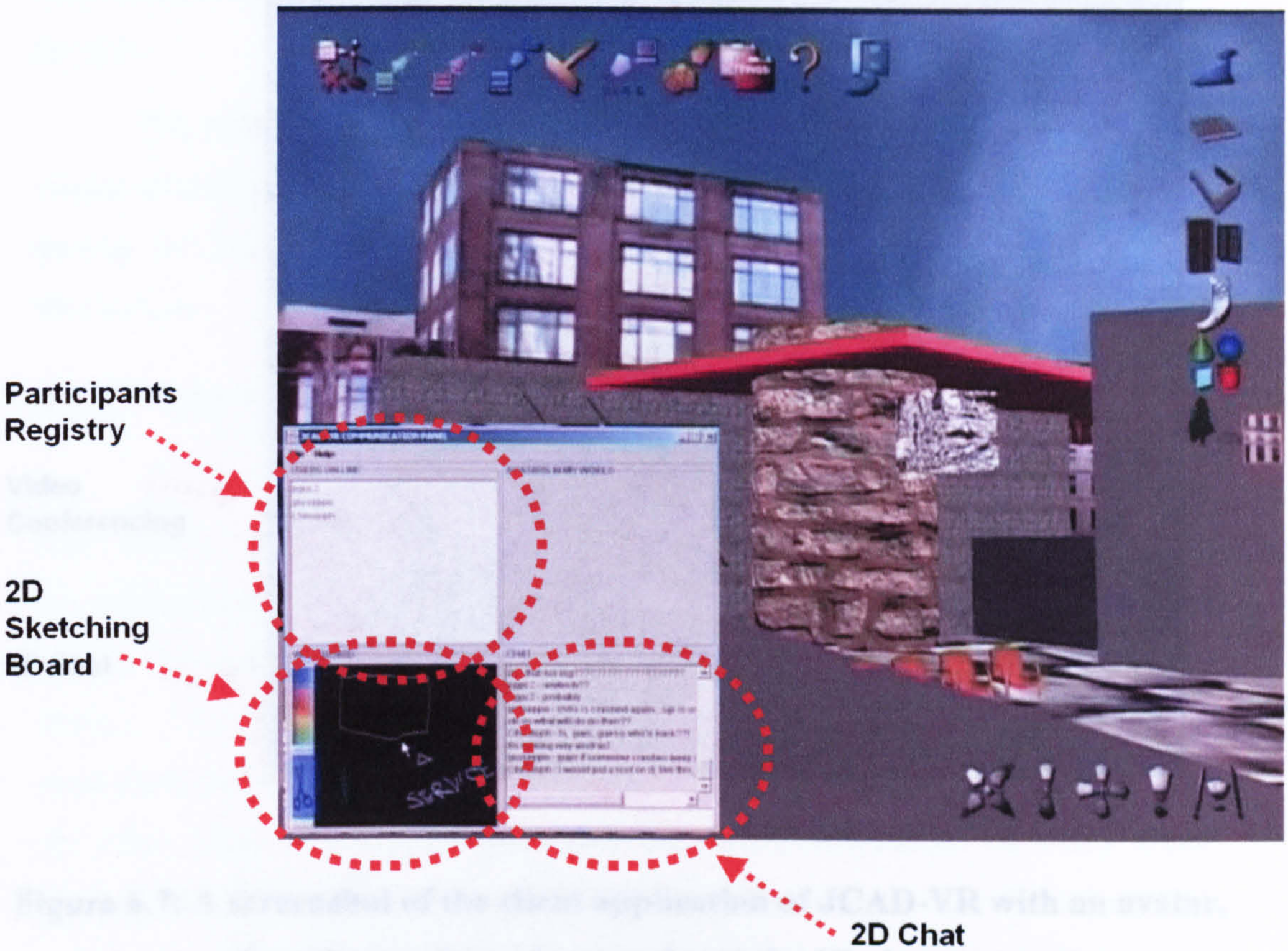


Figure 6.6: A screenshot of the client application of JCAD-VR with the tools available in the communication panel: a participants' registry, a whiteboard for sketching in 2D and a chat board

Figure 6.7 shows another screenshot of the client application of JCAD-VR with the 3D video conferencing panel, an avatar with its username and the 3D chat area. These are parts of the 3D interface of the VE therefore their detailed descriptions will not form part of this thesis (See Section 6.5 of the companion thesis Conti, 2002).

The following section will describe in detail the *Sharing* core of the system which is responsible for the streaming of geometrical information related to the Virtual Environment.

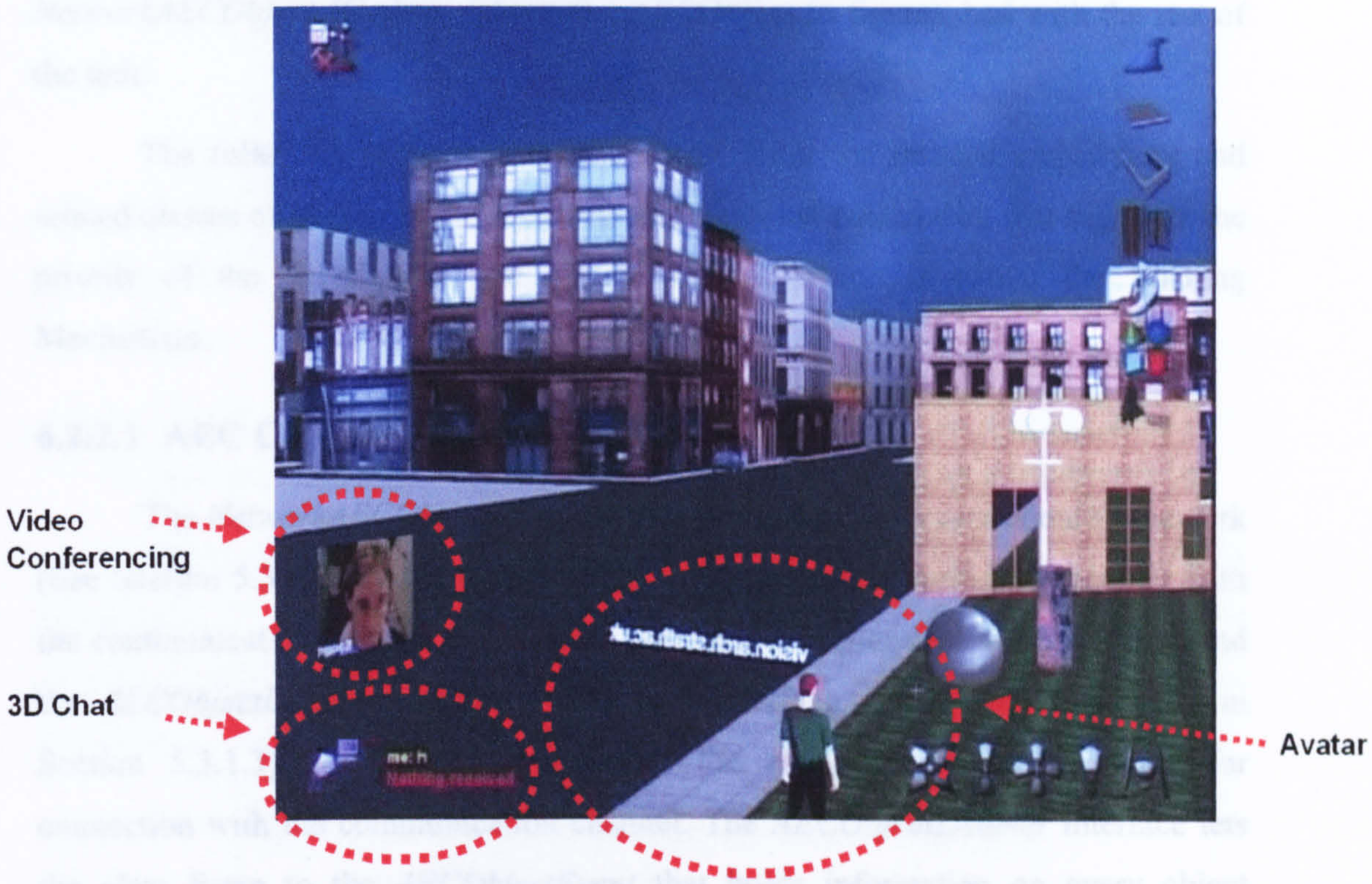


Figure 6.7: A screenshot of the client application of JCAD-VR with an avatar, the video conferencing panel and the 3D chat area

6.2.2 The Sharing Core

The *Sharing* core allows the streaming of information about the objects included in the virtual world and it also regulates the priority of the users and their interaction with these objects. Every user can create, modify and delete 3D objects while navigating the VE and so the particular responsibility of the *Sharing* core is the transmission of geometrical information about these objects. The *Sharing* core upgrades in a synchronous manner the virtual scene every time an object has been created or manipulated by the remote participants.

The most important class of the *Sharing* core is the *NetworkAECObjectsIO* class, which is directly responsible for the streaming of objects' geometrical information.

Figure 6.1 illustrated the general internal architecture of the client side of the JCAD-VR networked unit and it showed the relationship that the

NetworkAECObjectsIO class, (shortened AECObject in figure), had with the rest of the unit.

The following sections will describe in detail the internal architecture and related classes of *NetworkAECObjectsIO* and show the mechanism that regulates the priority of the users' over the virtual objects, which is called the Locking Mechanism.

6.2.2.1 AEC Objects

The *NetworkAECObjectsIO* class is registered to the server/client framework (See Section 5.3.1.3) by the *ClientInFrame* class (See Section 6.2.1) together with the communication tools of the *Network* core, and it implements the *ChatClient* and the *AECObjectListener* interfaces. The first interface, (as already mentioned in Section 5.3.1.3) provides the class with the message streams necessary for connection with the communication channel. The *AECObjectListener* interface lets the class listen to the *AECObjectEvent* that holds information on every object created, modified or deleted in the VE. More specifically every time the user creates, modifies or deletes an object from the virtual scene the relevant information has to be transmitted to the other participants in order to keep the VE upgraded. Moreover information about the object has to be in numerical format to be sent and received through the underlying network and so the development of an information structure able to describe every aspect of the object numerically was necessary. This data model translates the actions performed on the object, their type, material and dimensions, into numerical values providing a unequivocal geometrical definition of the object. In other words every object that is created, modified or deleted by any user is described through figures which entirely define its status, object type, material and dimensions. Figure 6.8 shows how object's parameters, except dimensions, are translated into predefined numerical values.

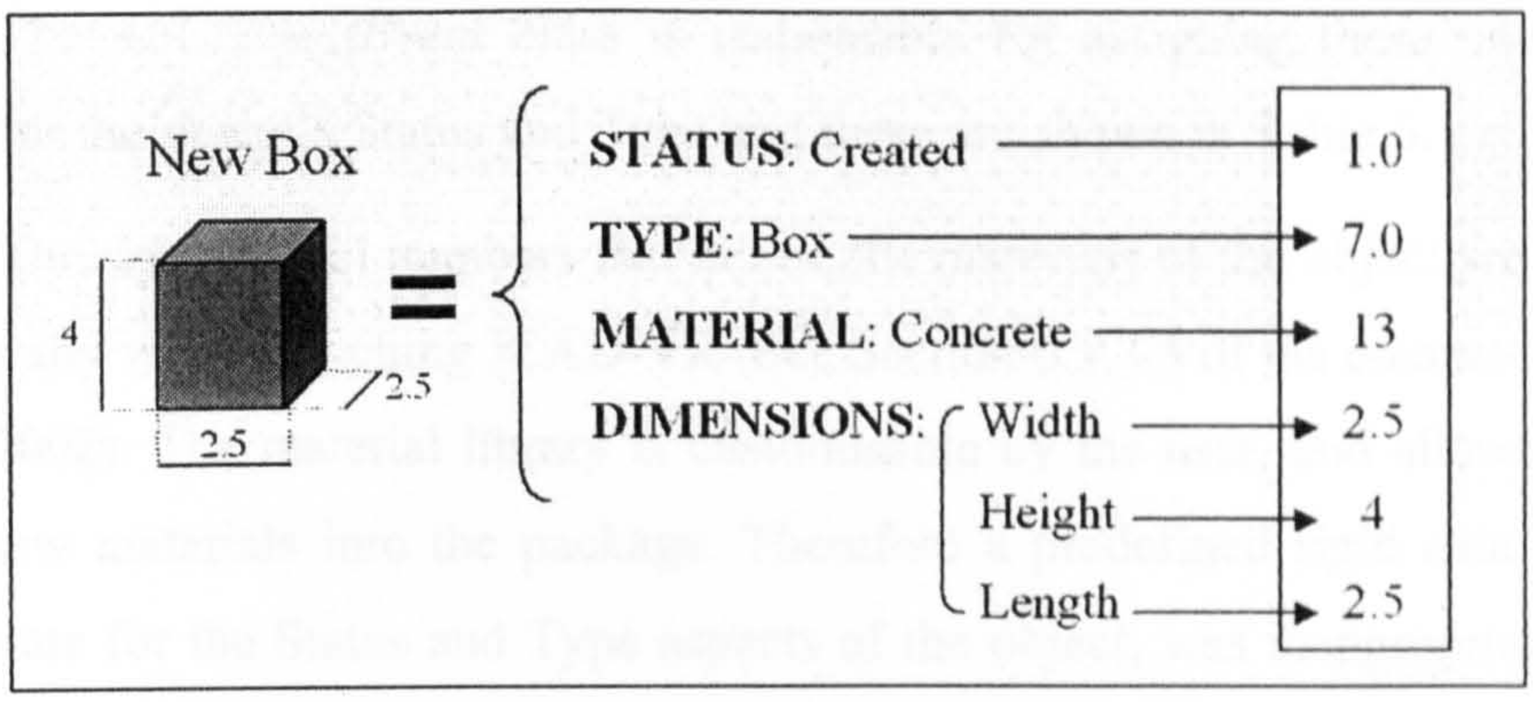


Figure 6.8: An object's parameters in JCAD-VR defined numerically

Figure 6.8 shows the expression of the different parameters of an object numerically, for example, if this new box had been just created, its Status value would be 1.0 (1.0=created), its Type value 7.0 (7.0=box), Material value 13 and its Dimensions would be expressed in meters.

Status	Type
Created = 1.0	Room_Generic = 2.1
Changed = 1.1	Wall_Default = 3.0
Changed_Material_Only = 1.15	Slab_Square = 3.5
Locked = 1.2	Door_1 = 4.0 ↓ Door_5 = 4.4
Unlocked = 1.3	Window_1 = 5.0 ↓ Window_8 = 5.7
Refreshed = 1.4	Sphere = 6.0
Deleted = 1.5	Box = 7.0
	Cylinder = 9.0
	Cone = 10.0
	Stair_1 = 501 ↓ Stair_6 = 506
	Object_1 = 1000 ↓ Object_24 = 1023

Table 6.1: The data model in AECObjectEvent

The *AECObjectEvent* class is responsible for assigning these unequivocal figures for the object's Status and Type and these are shown in Table 6.1.

The unequivocal numbers that define the materials of the object are assigned dynamically when launching JCAD-VR (See Section 6.5.3.5 of the companion thesis Conti, 2002). The material library is customisable by the user, and allows them to insert new materials into the package. Therefore a predefined rigid data structure appropriate for the Status and Type aspects of the object, was inappropriate for the Materials.

Once all the object's parameters in numerical format have been obtained, the data has to be arranged in a structured manner in order to represent the object correctly. Consequently the object's parameters are stored in an array of figures with a predefined structure and length that is able to represent any type of object included in JCAD-VR. In particular every array defines an object and every position of this array provides numerical information about a predefined parameter of the object. Table 6.2 clarifies the structure of the array, showing which parameter is being described by each of its positions.

Within JCAD-VR then every object is defined by an array of 76 positions that can be easily transmitted through the communication channel to the remote participants in order to upgrade the virtual scene. In many cases the object does not need 76 parameters in order to be properly defined therefore some of the positions of the array may contain null values.

Note that positions 26 to 31 are actually there to allow storing any parameter which would be useful for the implementation of further tools.

The information about the object is stored in the array during the interaction with the VE (See Section 7.3 of the companion thesis Conti, 2002). In particular, every time a virtual object is created, modified or deleted its array is stored in a *AECObjectEvent* class and this is sent in the form of an event to the other internal classes of JCAD-VR (See Section 7.4 of the companion thesis Conti, 2002).

POSITION	CONTENT
0	**Status:** Shows if the object was Created, Changed, Locked etc.
1	**Type:** Defines the type of object i.e. box, sphere, cylinder, wall etc.
2-13	X,Y,Z coordinates of the four points which define the majority of the primitives
14	Total volume of the object
15-16	Materials
17	Width
18	Height
19	Length
20	Internal Surface
21	External Surface
22	Total Surface
23	Radius
24	Radius 2
25	ID of the parent wall for applying windows and doors
26-31	Empty
32-63	Transform3D (Java3D™)
64-75	Parameters for modification of walls

Table 6.2: The composition of the array with every position occupied by a predefined type of information about the object

The *AECObjectEvent* class holds another important piece of information about the object. This is its identification number, called the *Key*. The key of an object, or of its array, allows quick identification of the different objects present in the VE and makes the routines of storing and retrieval from objects containers (such as vectors) easier. Therefore it is possible to easily identify one relevant object among many others through its unique key. Since the key has to be sent through the network with the object's other parameters, although it is a number it has to be unique for each of the objects present in the shared virtual scene. To ensure uniqueness throughout the networked system the key number is defined by an algorithm that combines the client's unique ID number which was assigned by the server after connection (See Section 5.5.1), and a shape ID number, called

keyDouble, that is assigned to every object at its creation (See Section 7.4.3.1 of the companion thesis Conti, 2002).

Once a key number and the array for the object are defined, the *NetworkAECObjectsIO* class is responsible for sending and receiving these arrays by using message streams (See Section 5.3.1.1) and then for transmitting retrieved information to the VE. Further before sending any message to the network, *NetworkAECObjectsIO* extends the object's array by adding one more position to the initial 76. This position allows the storing of the object's key number and therefore the final array that is transmitted to the communication channel finally looks as it does in Figure 6.9.

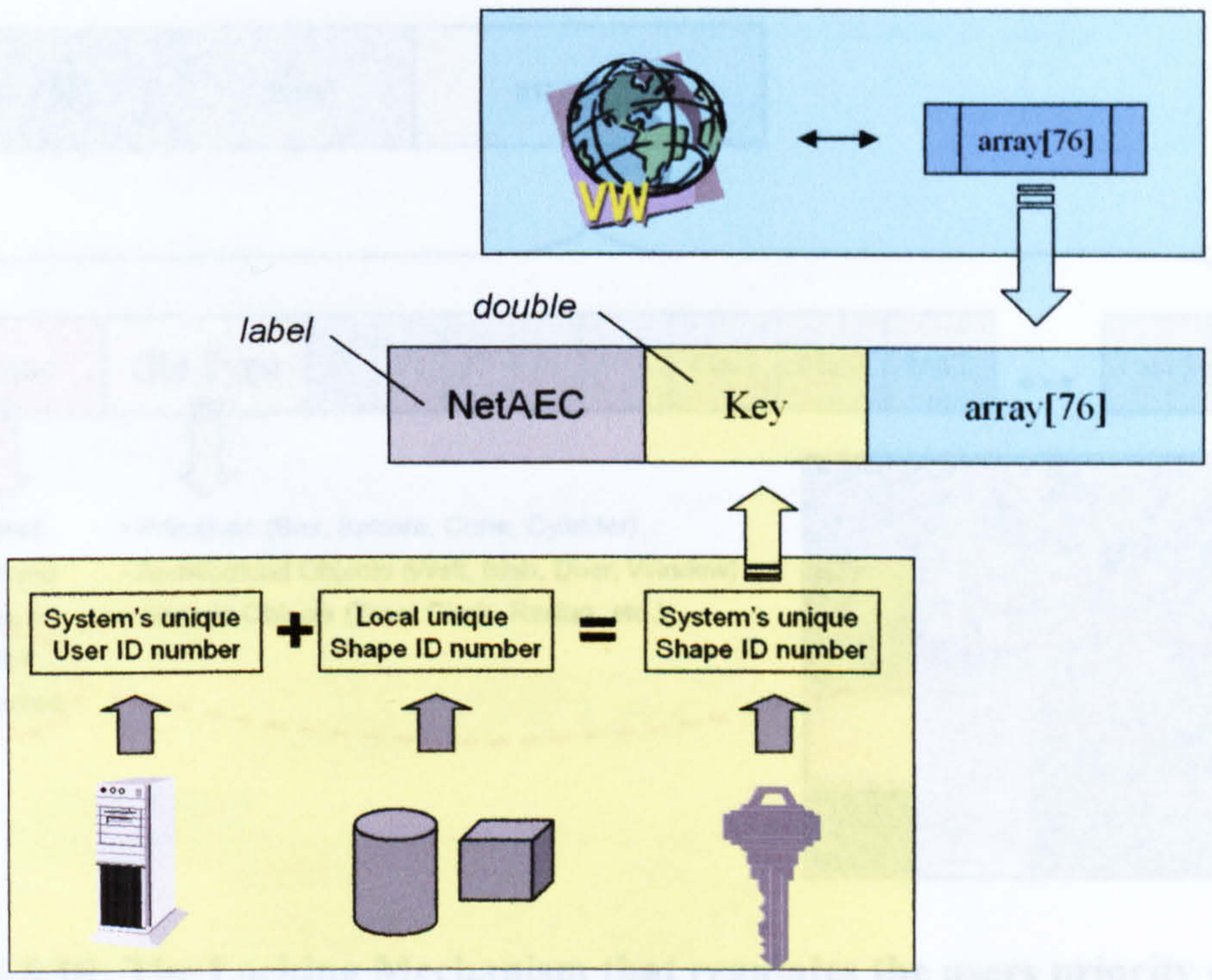


Figure 6.9: The array sent and retrieved through the communication channel has the label indicating the tool for multiplexing, the key that identifies uniquely the object and the array of 76 positions

Figure 6.9 shows the array sent and received from the communication channel. It shows the label for the multiplexing routine indicating the AECObject tool (See Section 5.3.1.2), the key that uniquely identifies the object and the array of 76 positions that defines the geometry of the object.

6.2.2.2 Locking Mechanism

The Locking Mechanism regulates the priority of the users over the virtual objects. This is activated any time a user selects an object in the VE and it results in the object turning red in the virtual scene of the remote participants, and the impossibility of it being edited by any other user apart from the one who selected it first. The concept behind its implementation is fairly simple and is strongly related to the value of the Status of the object. The structure of the information model of the virtual objects was described in Section 6.2.2.1 and among the parameters Status was the one responsible for notifying if the object had been Created, Changed, or Deleted etc. but especially if it had been Locked or Unlocked.

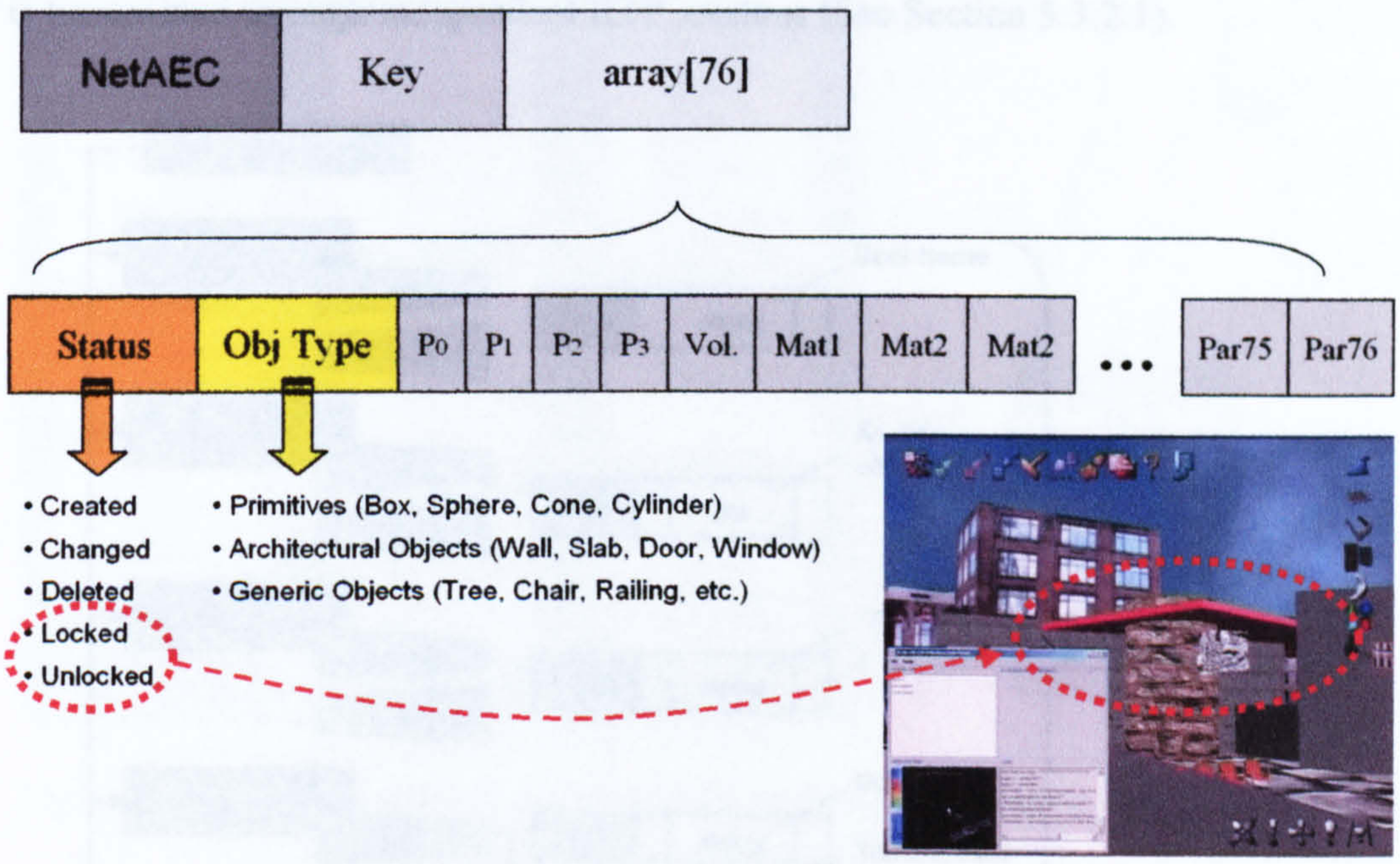


Figure 6.10: The Locking Mechanism that regulates the users priority over the virtual objects

Figure 6.10 illustrates this mechanism with the array carrying all the information about the object, and also contains information about the Locked or Unlocked status. The change from one status to another is regulated by the interaction procedures with the VE (See Section 7.4.3.1 of the companion thesis Conti, 2002). If the object's Status is Locked, for the classes dealing with the management of the VE, this means it is impossible to edit the object and it will have an automatic change of colour to clearly express its status to the user. If the object is

then released or Unlocked, as soon as it is deselected by the first user it will return to its initial material in the remote VEs and it will be available for selection again.

6.2.3 Coexistence of Tools

The *Network* and *Sharing* cores embrace all the classes that allow collaboration in the client application of JCAD-VR. In particular, the *Network* core provides classes that focus mostly on "human" communication among the participants and the *Sharing* core focuses mostly on the transmission of geometrical information for the virtual world. Figure 6.11 shows a table with most of the tools available in JCAD-VR with their arrays filled with the data relevant to each tool. Note that the video conferencing facility has not been included in this figure since it is transmitted through independent RTP sessions (See Section 5.3.2.1).

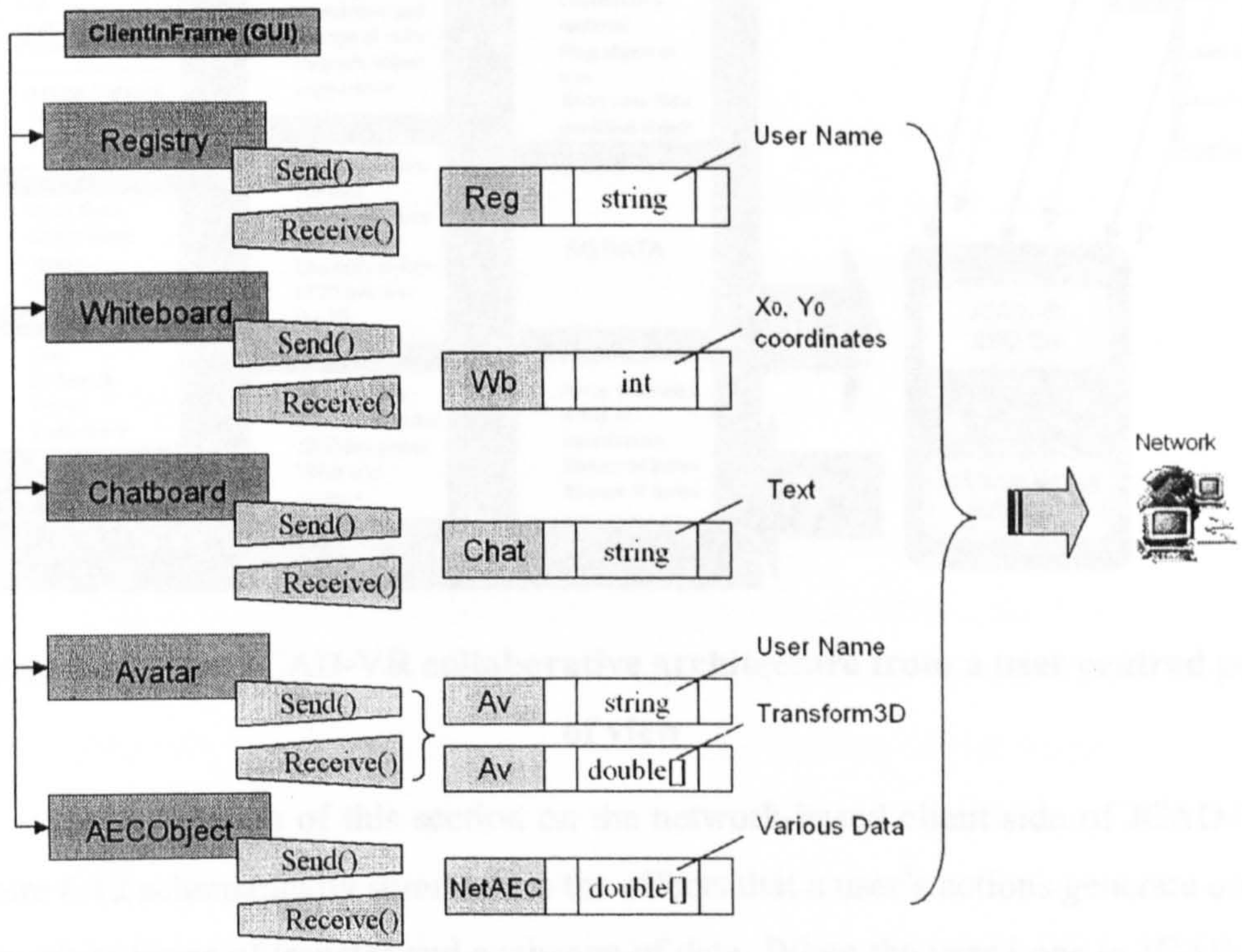


Figure 6.11: The coexistence of different tools in JCAD-VR is achieved using Multiplexed streams

Figure 6.11 shows the advantage of using Message Streams (See Section 5.3.1.1) and especially in using Multiplexing and Demultiplexing mechanisms in

networked applications (See Section 5.3.1.2). These streams allow the coexistence of various collaborative tools by using a unique communication channel and by so doing they make the implementation of the network-based unit significantly easier. All the tools were developed independently and their implementation did not require any special work in order to avoid conflict during transmission.

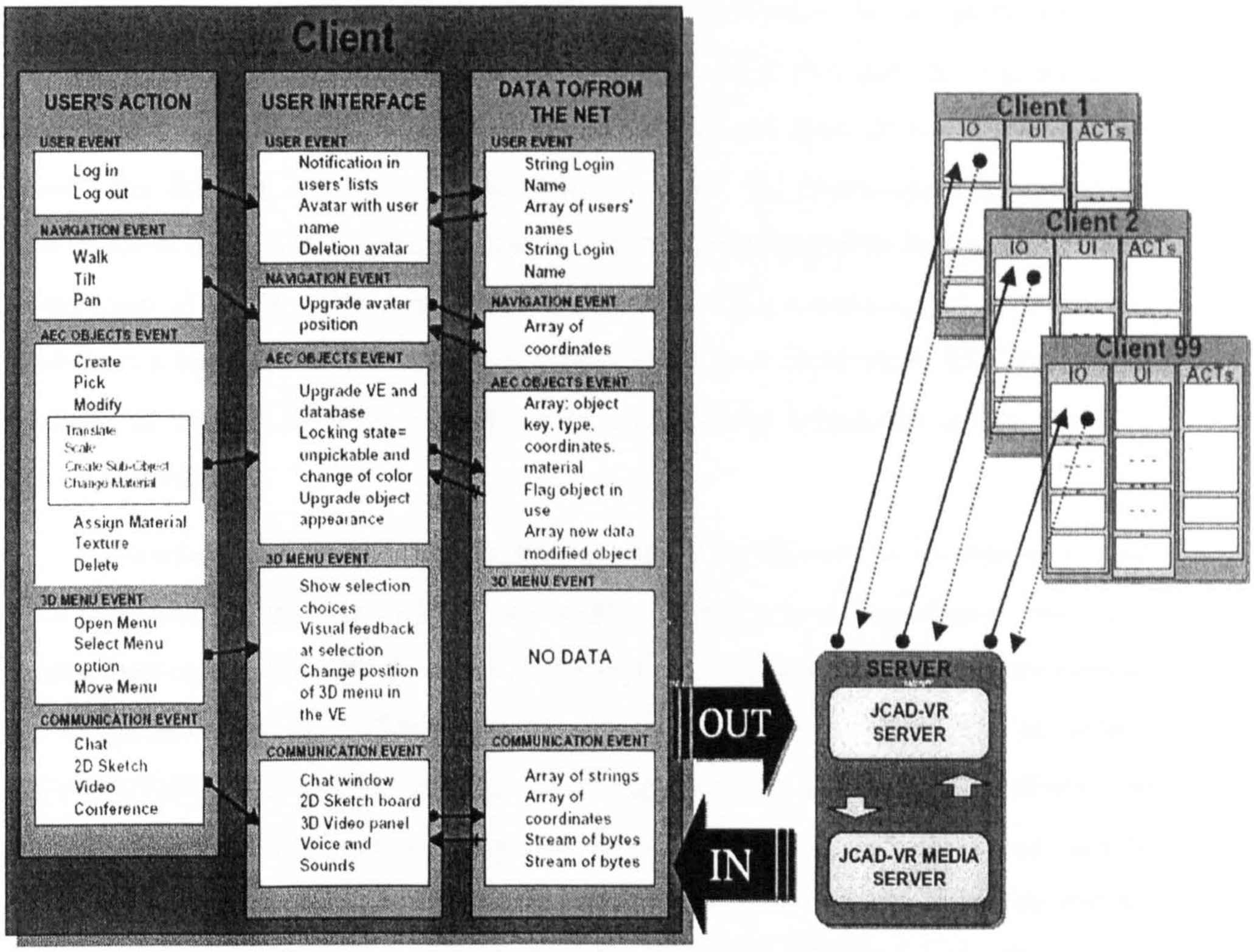


Figure 6.12: The JCAD-VR collaborative architecture from a user centred point of view

In conclusion of this section on the network-based client side of JCAD-VR, Figure 6.12 schematically summarises the effects that a user's actions generate on the network in terms of transfer and exchange of data. When the user Logs in JCAD-VR this has a visible effect in the visualisation of the username by the *RegistryABACUS* GUI and the creation of the new user's avatar in the VE. In terms of networking, the effect is the transmission of the user's Login name to the server and the reception from the server of the list of usernames. Finally the figure illustrates the visual

feedback resulting from actions that may be performed by the user and it shows the type of data that is consequently transferred to the communication channel.

6.2.4 Network Persistence

A Virtual Environment created within JCAD-VR exists in the memory of the computers that are involved in the collaborative session until the last participant has left the session. As already mentioned, the server does not have the capability of reading and retaining data coming from the clients and then of storing the virtual scene (See Section 5.5.1). From this point of view the networked architecture of JCAD-VR is limited as it does not offer a centralized information repository. Further in the case of a new client connecting midway through a session the server cannot send them a copy of the created virtual scene resulting in the dramatically unpractical scenario of waiting for all the clients to be successfully connected before a session can be started.

To avoid this inconvenience the system delegates one of its clients to deal with this issue. Thus every time a new client connects to a commenced session, a mechanism called Network Persistence offers a way to transmit a copy of the created VE to the new participant. Every time a new participant joins the session, as already mentioned the *RegistryABACUS* class of all the clients creates a new *UserEvent* and it passes information about the status of the new user, e.g. their username and ID number (See Section 6.2.1.3). *UserEvent* holds a list of clients IDs internally and the new ID is stored or deleted (in the case of them leaving). Within the *UserEvent* class a routine checks if the ID number of the local client is the lowest on the list. This routine requires the identification of one client among those connected as the responsible client for the transmission of the VE to the new participant. Each of the clients holds this list of IDs locally and each of them makes checks concurrently. Finally the client with the lowest ID number saves the objects in the VE in a file (See Section 6.2.5.3), deletes the whole environment and then reloads the environment using this saved file (See Section 6.2.5.3). This procedure is performed in few seconds and it allows the creation of a replica of the objects present in the scene and

their transmission to the other participants, including the one that joined the session last.

It is less efficient in terms of networking usage than the centralized option with a server holding information on the virtual scene and dispatching the virtual objects to the sole joining participant, however this mechanism ensures access to the collaborative session at any time and in case of the network failure of one or more of the clients it offers the possibility of keeping the virtual scene alive.

The last part of this chapter is the description of the *Data* core that manages the clients' internal database and the loading and saving routines of the system.

6.2.5 The Data Core

The client application of JCAD-VR includes the *Data* core that manages the database internal to every user's computer, it allows input from CAD/CAAD packages and it performs save and load routines to/from a proprietary file format called *jcad*.

The following three sections will cover these topics and will describe the internal architecture of the module.

6.2.5.1 The Internal Database

This is a local database that stores all the objects present in the virtual scene in an internal data container and then allows the users to see its contents. Its structure consists of four classes, the *DataAECCollector* and the *DatabaseManager* with its GUI classes, the *DataAECInterface* and the *JInternalFrameAECObject* class. The first manages the routines for storing the objects by type while the second organises the set of data to be visualized in a 2D panel called *object viewer*. This viewer is implemented in *DataAECInterface*, which calls on the *JInternalFrameAECObject* class to show the user information about each of the object types.

The *DataAECCollector* class implements *AECObjectListener* and *AECNetworkedObjectListener*. The last listens to the *AECNetworkedObjectEvent*

class that is fired by the *NetworkAECObjectsIO* class (See Section 6.2.2.1) every time a new object is received through the network. *AECNetworkedObjectEvent* extends the *AECObjectEvent* class (See Section 6.2.2.1) therefore it has the same structure. It was only included in the package to manage the difference between the events that were notifying the local creation of a new object from the events that were fired after the reception of a new object from the communication channel. In other words the *AECObjectEvent* class notifies when the user creates a new object locally while *AECNetworkedObjectEvent* announces that objects have been created by remote participants and are being received through the network.

DataAECCollector is primarily responsible for the creation of object containers for each of the object types included in the JCAD-VR system. These containers, which are called *Maps* or *associative arrays*, hold key-value pairs of objects (Eckel, 2000) and they can be seen as small databases where it is possible to store and retrieve information through key entries. The *DataAECCollector* class checks for the type of object, e.g. box, sphere etc., and for the status of the object, e.g. created, deleted etc., and accordingly stores or deletes the object in the appropriate map. Figure 6.13 illustrates the general mechanism, showing the array (held either by *AECObjectEvent* or *AECNetworkedObjectEvent*), checked against its object type and the key and its relative array being stored or deleted from the appropriate object's map.

The *DatabaseManager* class is a service class that sorts the information contained in the object's array into the right format before it is visualised in the *object viewer*.

As mentioned, the *DataAECInterface* and the *JInternalFrameAECObject* classes hold the GUI of the internal database. In particular, the *DataAECInterface* class provides a 2D window where the user can retrieve numerical information about all the objects that are stored in the various maps. In particular, this class is responsible for creating a frame that contains a handler of the *JInternalFrameAECObject* class for each type of object allowed by JCAD-VR. More specifically, as shown in Figure 6.13 every handler of the *JInternalFrameAECObject* class is responsible for the visualisation of numerical information about one object's

type only, with the data held by the object's array that was previously sorted by the *DatabaseManager*.

The result is a simple presentation tool that allows the user to monitor all the objects created in the virtual scene and also provides the means for the fast retrieval of numerical information otherwise not easily accessible.

Even at this initial stage this tool, paves the way towards a system that will allow the integration of 3D modelling and visual simulation with control over the numerical parameters of the model.

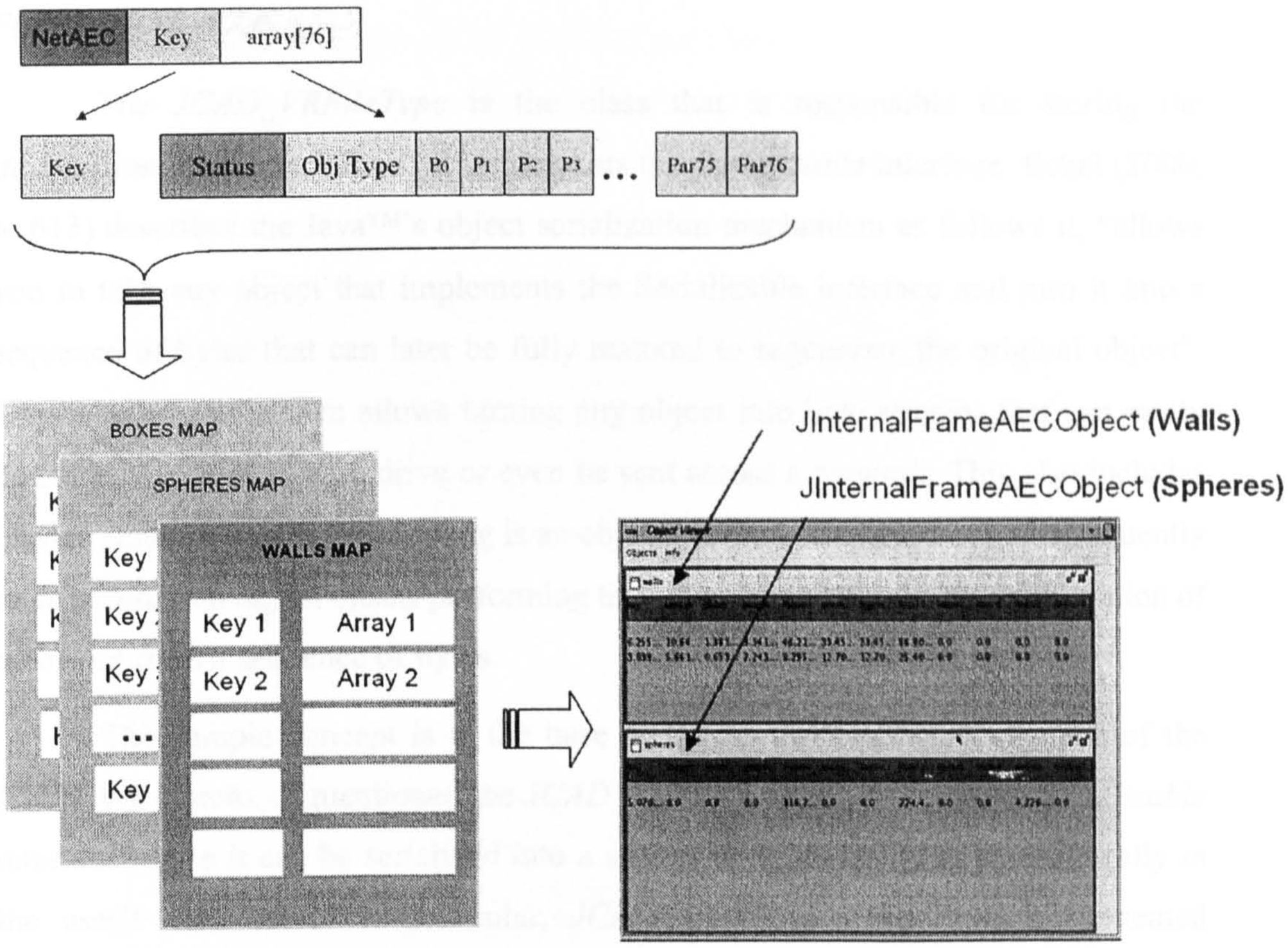


Figure 6.13: The internal database holds information about all the objects present in the VE and visualises their data in a 2D window

6.2.5.2 Input from CAD/CAAD

JCAD-VR allows the user to load pre-existing 3D models to form a background context for their design. These environments can be built using standard CAD/CAAD packages and can be simply loaded into the JCAD-VR system as

VRML files. The user can then create new objects and primitives conveniently within the environments which have been loaded. See Section 7.6 of the companion thesis (Conti, 2002) for a detailed description of the use of this facility.

6.2.5.3 Save/Load jcad Files

JCAD-VR allows saving and loading of a proprietary file format for the convenience of the user. The user can save the virtual objects created during a collaborative session to the hard drive and then load them at their convenience.

This routine is evident in two main classes: *JCAD_VRFileType* and *FileDialogABACUS*.

The *JCAD_VRFileType* is the class that is responsible for storing the information about the VE and it implements the *Serializable* interface. Eckel (2000, p. 613) describes the Java™'s object serialization mechanism as follows it, "allows you to take any object that implements the Serializable interface and turn it into a sequence of bytes that can later be fully restored to regenerate the original object". Object serialization then allows turning any object into byte streams that can easily be stored in the local hard drive or even be sent across a network. This also includes classes since in Java™ "everything is an object" (Eckel, 2000, p. 101). Consequently to *deserialize* an object means performing the opposite procedure: the regeneration of an object from a sequence of bytes.

This simple concept is at the base of the saving/loading mechanism of the JCAD-VR system, as mentioned the *JCAD_VRFileType* implements the *Serializable* interface hence it can be serialized into a stream of bytes and then stored locally in the user's hard drive. In particular, *JCAD_VRFileType* has been implemented because of its function as a data receptacle. It holds four *vectors*, or object containers, each storing the following elements:

1. The arrays of the virtual objects, e.g. boxes and walls
2. The keys of the virtual objects
3. The arrays of the virtual sub objects, e.g. doors and windows included in a wall

4. The paths of the VRML files that are loaded as contexts of the VE.

These four vectors cover all the relevant aspects of a VE and being part of the *JCAD_VRFileType* class, once serialized they store a copy of its content in the local hard disk.

The *FileDialogABACUS* class is responsible for providing data to fill the vectors and it performs the procedures of serialization and deserialization for the handlers of the *JCAD_VRFileType*. More specifically *FileDialogABACUS* retrieves all the virtual objects, sub objects and the paths of the VRML files from the internal classes of the VE (See Section 7.2.1 of the companion thesis Conti, 2002). It then calls up a new handler of the *JCAD_VRFileType* class, passes information about the VE to it and finally it activates the serialization procedure. Concurrently, the new handler stores the data in the appropriate vectors and gets serialized in a *jcad* file, whose path is defined by the user through the save window as shown in Figure 6.14.

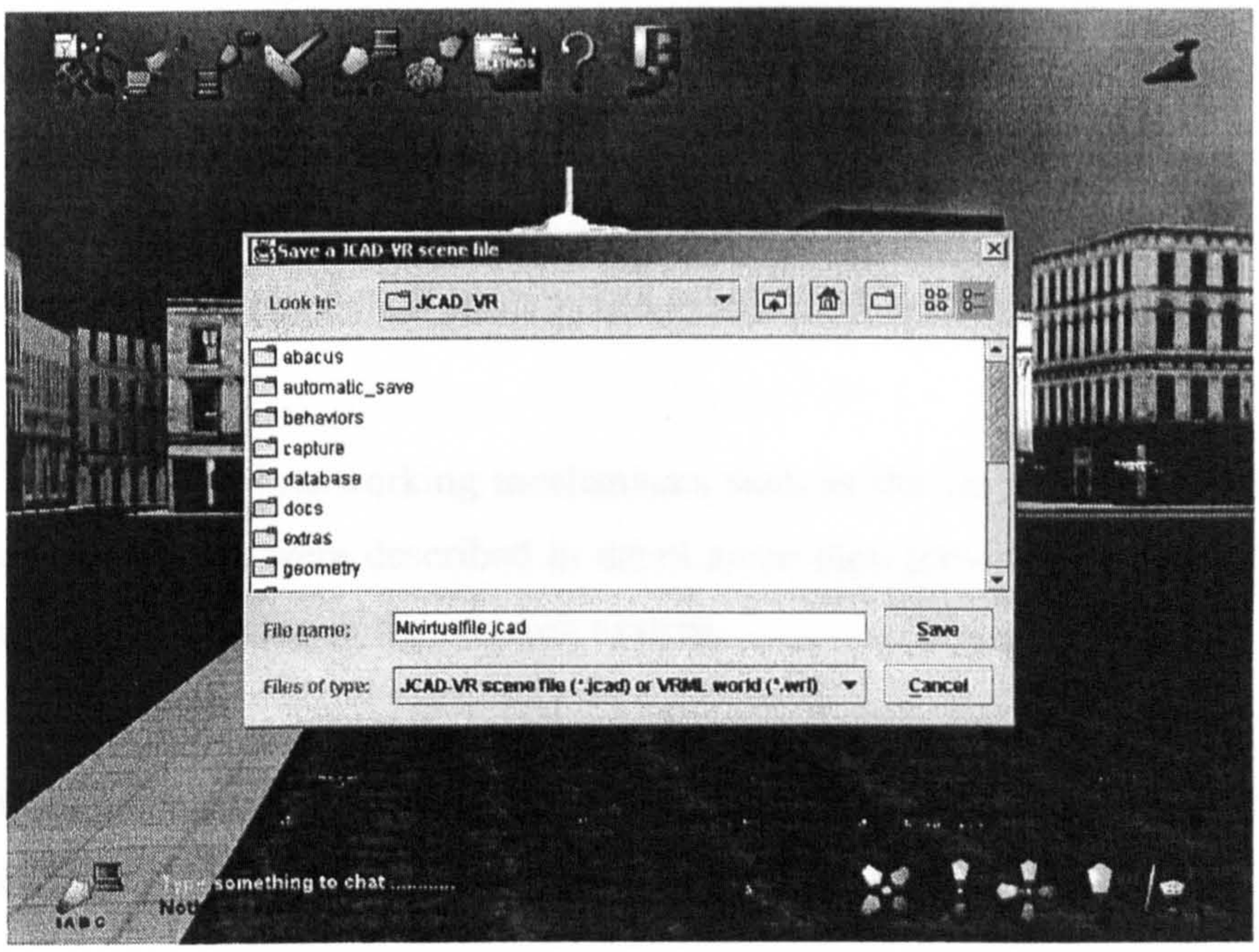


Figure 6.14: A screenshot of the save/load 2D window of JCAD-VR

Finally the loading procedure is based on the deserialization of the wanted *jcad* file. In detail, *FileDialogABACUS* deserializes the handler of the *JCAD_VRFileType* class, accesses the vectors and starts decoding the data in order to reconstruct the VE. First it decodes the paths of the VRML files and passes this

information to the routine that imports the files (See Section 7.6 of the companion thesis Conti, 2002), then it decodes objects, keys and sub objects that are eventually passed to the *AECObjectCreator* class that will recreate the virtual objects in the VE.

Note that the user will be able to reload the context of the VE only if the appropriate VRML file is placed in the same location which was stored in the vector of the file paths, because the vector that holds the contexts does not store the VRML files but only their file paths.

6.3 Conclusions

This chapter described in detail the internal architecture of the client side of the network based unit of JCAD-VR. First it gave an overview of the client unit and then it described in detail the three cores of the client application: the *Network*, *Sharing* and *Data* cores.

It also provided a description of their internal classes and their main routines clarifying, when necessary, their links to the VE and providing references to relevant information in the companion thesis that together with this work forms the JCAD-VR project.

Lastly several networking mechanisms such as the Locking Mechanism and Network Persistence were described in detail since they provide the means for data consistency and stability in the network system.

The following Chapter 7 will address the testing phase of the JCAD-VR system. In particular it will report on an experiment that used JCAD-VR and involved students in accomplishing a real design task.

7 Testing JCAD-VR: a Collaborative Session

7.1 Introduction

The three previous chapters highlighted the most important aspects of the JCAD-VR framework and described its internal architecture in detail. This chapter will report on the testing of the functionality of the software. An actual collaborative scenario was established to evaluate the effectiveness and ease of use of JCAD-VR in practise and the following sections will describe the experiment in detail.

The following section will also outline the *JCAD-VR TU/e Version* of the software. This is the version of the prototype specifically customised for use in this experiment. This version features a number of mechanisms for monitoring and recording the progress of the design and communication processes.

7.2 The JCAD-VR TU/e Version

The *JCAD-VR TU/e Version* of the software was originally designed for a collaborative experiment between the Technische Universiteit Eindhoven (TU/e) in the Netherlands and the University of Strathclyde in Glasgow (See Chapter 8).

This version of the client application is enhanced with several features that help to monitor both the progress of the design and the information flow among the participants. In addition to the tools available in the standard JCAD-VR system, which have been outlined in the previous chapters, this version also features a number of routines specifically developed for the testing of the software, providing in particular:

- Automatic loading of a 3D-model of the context area chosen for the design task
- Automatic saving of the content of the VE
- Automatic capturing of screenshots during the session
- Automatic recording of the content of the communication by chat.

The first feature makes the setting up of the environment for the collaborative session easier. As soon as the system is started a routine automatically loads the VRML97 file that was specifically chosen as the context for the design task rather than the user being asked to choose a file. This routine performs the loading procedure (See Section 7.6 of the companion thesis Conti, 2002) automatically. The participants are however free to load further 3D-models of virtual contexts by clicking the "load" 3D-icon and selecting the preferred VRML97 files.

The remaining three routines were developed to track the evolution of the design process throughout the experiment. The second routine specifically performs an automatic saving of the content of the VE in *jcad* files (See Section 6.2.5.3) every 10 minutes. The files are then automatically stored in the "JCAD-VR\automatic_save\" folder with sequential numbers assigned (e.g. *file 1, file 2, file n*). In case of application failures this helps to prevent the loss of data and gives the ability to keep track of the progress of the design.

The third routine provides the means to progressively visualise the stages of the VE over certain intervals of time. Every 10 minutes this routine saves a screenshot of the scene in a *jpeg* file that is stored in the "JCAD-VR\automatic_save\" folder with a sequential number (e.g. *screenshot 1, screenshot 2, screenshot n*).

The fourth routine records the content of the communication through the chat and saves it automatically in a text file placed in the same directory as before. This allows the recording of the flow of information among the participants and it is a useful tool for checking the dynamics of the communication.

7.3 The In-House Collaborative Session

The *JCAD-VR TU/e Version* system was tested in an experiment located at the faculty of Architecture of the University of Strathclyde in Glasgow. Three fourth year architecture students were selected for the experiment. They were all familiar and skilled in the use of CAAD packages but none of them had previous experience of Virtual Reality Aided Design (VRAD) systems. The students were first introduced

to the functions of JCAD-VR and then they were asked to participate in a collaborative session.

7.3.1 The Choice of the Configuration

It was decided to give the experiment a set of very precise and constrained conditions for a sample collaborative session. This required:

- The use of several different operative systems to test the multi-platform nature of JCAD-VR
- The simulation of a limited bandwidth
- The use of standard computer monitors as visualisation devices

Consequently two standard PCs, two SGI computers, one Onyx2 and one O2 were used for the experiment. Their detailed hardware and software specifications are included in Appendix D. Specifically the two PCs and the Sgi Onyx2 were used for the client applications and the Sgi O2 was used to run the server. To ensure the network consistency in the unlikely event of all the three clients accidentally quitting at the same time (See Section 6.2.4) the Sgi O2 was used to run a further client application remotely on the Onyx2.

In addition, in order to simulate having only a limited bandwidth available, the communication was restricted to the use of the chat and whiteboard tools. Furthermore, as illustrated in Section 4.6.1, due to the limited functionality of the Java™ Virtual Machine which was available for the Irix OS, JCAD-VR's videoconferencing module was not available on the Sgi computers. Therefore the decision not to use the videoconferencing module made it possible to use Sgi machines and at the same time provide all the students with the same tools. However, the video conferencing module had been informally tested during the introductory session, as shown in the video provided with this thesis (See Appendix C).

Finally it was decided that none of the students should use the Reality Center™ at the ABACUS Unit. As explained this was justified, by the purpose of the test which aimed to reproduce a realistic daily working scenario where architects

would be interacting from several different locations by using standard monitors as visualisation devices.

It is predictable however that the efficacy of JCAD-VR would improve if either the videoconferencing module or the Reality Center™ were used, although this has not yet been rigorously proven by further formal experiments.

7.3.2 The Training Session

An introductory session of only twenty minutes duration was held to show the students the features of JCAD-VR and to illustrate the tools accessible to them from its 3D-GUI.

The students were already aware of the basic concepts of VR and had some experience with Desktop-VR. As mentioned earlier all of them were skillful in 3D modelling. However, none had previous experience in using either VRAD systems or collaborative applications and none had used the JCAD-VR software before.

During the introductory session students showed immediate interest and quickly demonstrated that they comprehended the simple logic behind the 3D-GUI. After the short introduction they all proved to be quite confident with the system.

7.3.3 The Design Task and the Outcome of the Experiment

The three students were located with their own workstation (See Appendix C) in three different rooms within the Department of Architecture. Fixed and hand-held video cameras constantly filmed their behaviour (See Figure 7.1).

None of the students were previously aware of the design task being set before they were informed at the beginning of the experiment. In this way the students did not have the chance to previously agree on design choices. Instead they were forced to only use the communication tools available within JCAD-VR to discuss their possible design solutions.

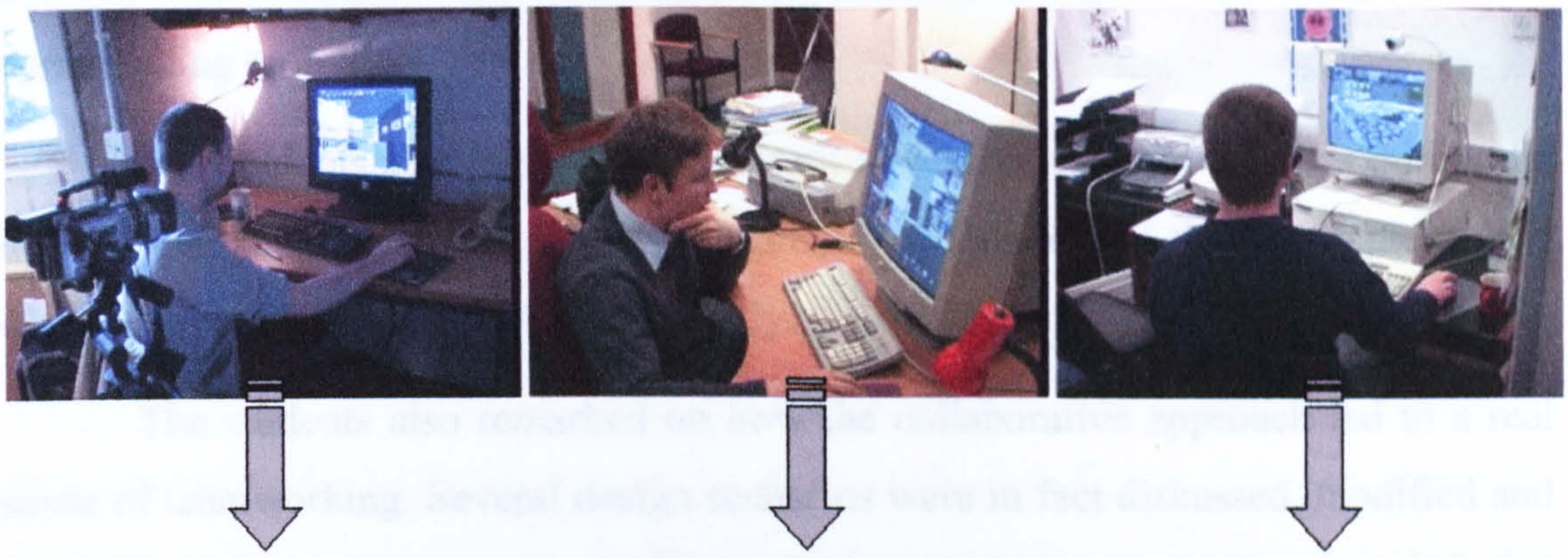


Figure 7.1: Images of the experiment with the students working on PCs and an SGI Onyx2

The three students were invited to design an information centre over the two-hour design session. The context area was already provided and it was loaded automatically by the system when it started. It consisted of a public square adjacent to a park.

Observing the video made during the session (See Appendix C) and studying the content of their communication through chat (See Appendix A) revealed several noteworthy aspects of the experiment.

First of all the system proved to be quite easy to use. Students did not have any particular difficulty in working with JCAD-VR and were able to use the system immediately after the 20-minute long training session. As mentioned (See Section 7.4 of the companion thesis Conti, 2002) due to a fault in the Java™ Virtual Machine for Windows OS, navigation with the PCs resulted in a slightly more complicated arrangement than originally planned. This resulted in the two students who were using PCs having to click on the relevant icon to switch to navigation mode rather than pressing the central button of the mouse. However, both students got accustomed to this after a few minutes.

The experiment was characterised by intense interaction amongst the three design participants. The students could communicate efficiently through the chat (See Appendix A) although they all stressed the potentially positive effects of having

a more direct means of communication, such as through voice or by a video conferencing tool.

The students also felt it was important to be able to draw sketches during the design session on the digital whiteboard provided. This helped them to express their design ideas quickly, clearly and efficiently.

The students also remarked on how the collaborative approach led to a real sense of teamworking. Several design scenarios were in fact discussed, modified and agreed on within the common design environment. Interestingly the shared design system was a cause of satisfaction as well as frustration among the participants. It was noted that students frequently changed or modified their own design solutions as well as everyone else's. This raised an interesting issue: the students suggested the development of some form of priority routine through which each user could decide whether the object being created could or could not be modified by other users.

Finally from the technical point of view the experiment highlighted the lower stability of the system on the Sgi platform if compared to PCs. The Sgi version suddenly stopped running three times while the PC versions were stable for the entire duration of the experiment. This was the result of the lower efficiency of the Java™ Virtual Machine for Sgi and the higher level of optimisation reached on the PC platform. As already mentioned (in Section 4.6.1) a higher priority was given to achieving efficiency of the system for the PC platform. This choice was justified by the more widespread use of PCs compared to Sgi set-ups.

Figure 7.2: Screenshots of three different stages of the design during the experiment

In addition to the sequence of screenshots taken during the experiment (See Figure 7.2 and Appendix B), a video camera was running in each of the rooms

filming the students. The collaborative session was then fully documented by a video that was later edited in order to show the most significant moments of the experiment.

The video is included in the thesis in Appendix C. It shows firstly the introductory session when the authors introduced JCAD-VR to the students and then the most relevant moments during the collaborative part of the experiment. The video also illustrates the final outcome of the design task (See Figure 7.3) and shows the students comments on both the JCAD-VR software and the collaborative session that were recorded at the end of the experiment.

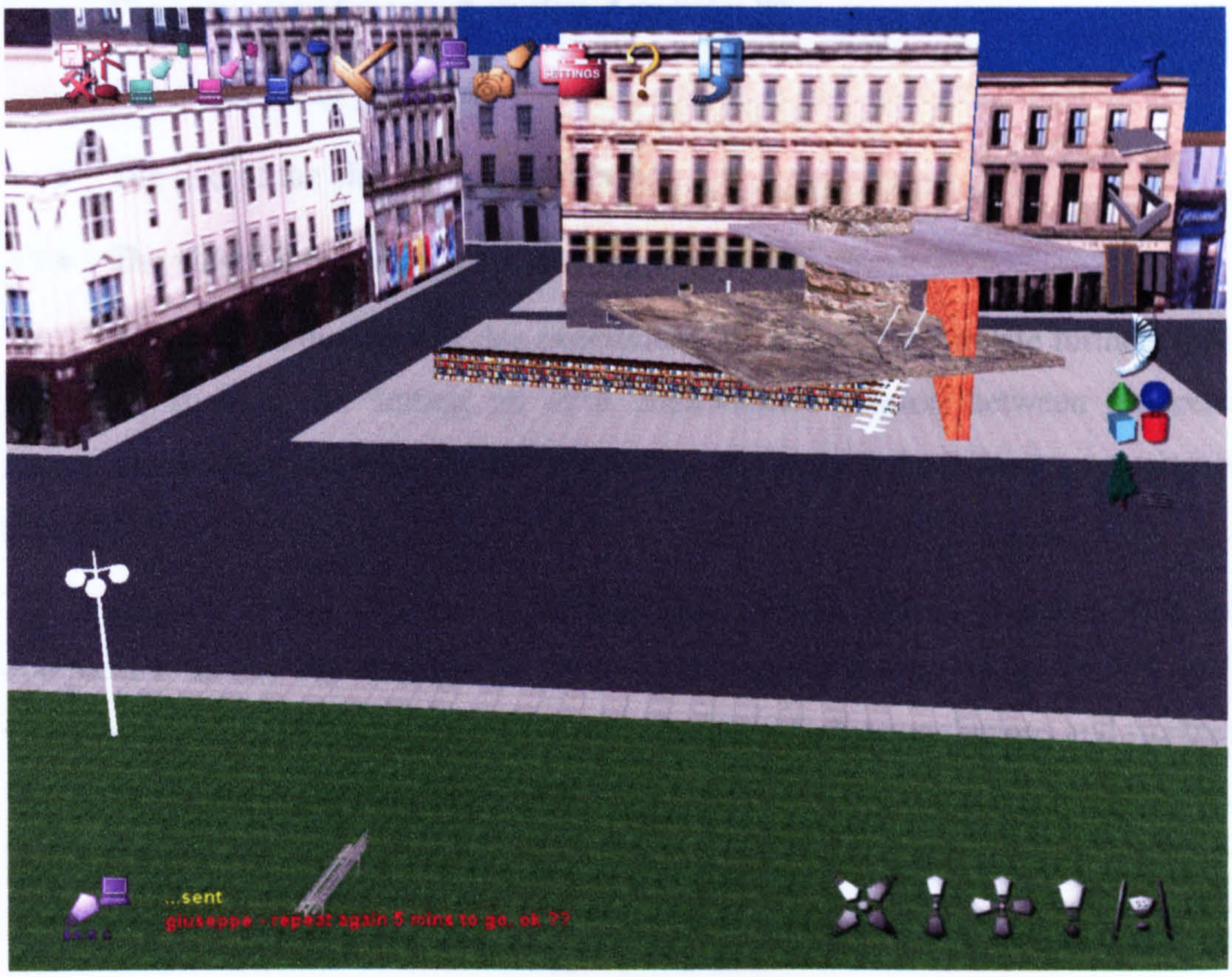


Figure 7.3: The final design of the collaborative session

From the students' interviews at the end of the experiment some suggestions emerged for further enhancements of the prototype. In particular they recommended the implementation of two other tools that would ease the dynamic of the collaboration: in their view a tool to help with the orientation in the VE, and a 2D map that would show from above the positions of all the participants in real time.

7.4 Conclusions

This chapter outlined the most important elements of the practical testing of the JCAD-VR system. The outcome of this limited experiment entirely proved that JCAD-VR, even at this prototype stage, is a fairly stable application that efficiently allows multiple participants to take active parts in a collaborative design session while communicating across the network.

Thanks to the specific features, which were implemented in the JCAD-VR TU/e Version, the experiment could be recorded in a number of ways:

- The sequence of screenshots (See Appendix B)
- The sequence of jcad files
- The text of the chat (See Appendix A)
- The video of the experiment (See Appendix C).

The positive outcome of this experiment will certainly stimulate further trials and tests, such as the setting up of a collaborative session between different institutions possibly between different countries, building on the in-house experience reported.

Chapter 8 will outline the contributions that this thesis, together with its companion thesis, has brought to the field of research into design in VR and Collaborative Virtual Environments. It will also provide the conclusions and suggest the future for the JCAD-VR framework.

8 Summary, Contributions and Further Work

8.1 Introduction

The previous chapter highlighted the methodology followed to test the JCAD-VR system, which provided useful feedback on the application.

This last chapter will provide a summary of the JCAD-VR project and will also outline the positive contributions that the development of the software prototype brought to the field of architectural design. This was achieved due to the development of the features typical of Virtual Reality Aided Design (VRAD) and Collaborative Virtual Environments (CVE) systems.

Conclusions to this work will be proposed along with plans for further work and developments to improve the system.

The section describing further works will in addition describe the methodology and preparation of an experiment at a much larger scale which will involve students from the University of Strathclyde in Glasgow, and the Technische Universiteit Eindhoven in the Netherlands.

8.2 Summary

This research project fulfilled four main tasks (See Figure 1.2). The first task was the research study that allowed the identification of the users requirements and also provided an overview of the basic characteristics of VRAD (See Chapter 3 of the companion thesis Conti, 2002) and CVE (See Chapter 3) systems, along with an attempt to categorise them into taxonomies.

The second task was the development of the JCAD-VR framework. This phase contained the following four sub-tasks:

- **Object Oriented Approach**. The choice of the OO approach is explained with separated modules to address different functions of the system and to allow

concurrent development of the system prototype. This led to the choice of Java™ as the programming language for the project.

- **Systems specifications and requirements.** The definition of the characteristics necessary for a collaborative VR system to allow synchronous collaboration and communication while designing in the VE.
- **Identification of the systems components.** The definition of the most important functions of the system and of its two main components, the VR design environment and the collaborative platform.
- **Separation of the competency in the two main themes.** The definition of the sub modules belonging to the VE and to the collaborative platform and the consequent separation of the competency. During this phase the final framework for the JCAD-VR prototype was defined.

The accomplishment of the third task was the implementation of the JCAD-VR system, and this was achieved through the combined efforts of the two authors. This was first done by choosing software packages and libraries which were compatible and by completing their integration into one application. Secondly, by the concurrent development by each of the authors of relevant modules of the framework and by checking their successful integration at the most crucial stages of the development.

The fourth task was the testing of the final prototype. An actual collaborative design session was conducted and documented using several different media. The results of the test were collected and reported in Chapter 7.

8.3 Contributions

JCAD-VR is an original application specifically designed to improve the architectural design process. The project embraces different aspects of the development of VR systems for collaborative design environments tailored for the conceptual design.

JCAD-VR brings several original contributions to the design process by allowing the earlier use of VR technology and by providing practitioners with an

effective and easy to use collaborative tool. Section 8.3.1 lists the advances in design methodology which were achieved by the implementation of JCAD-VR and its Human-Computer Interface. Section 8.3.2 illustrates the contributions offered by the development of both the system's collaborative platform and communication tools. Finally Section 8.3.3 outlines the original contribution that the implementation of JCAD-VR brought to the development of the first example of a new generation of Computer Supported Cooperative Work (CSCW) system (See Section 2.4) called the Collaborative Virtual Design Environment (CVDE) (See Section 3.4.5).

8.3.1 Effective Design in VR

As mentioned in Section 3.2 of the companion thesis (Conti, 2002) traditional CAD systems are not suitable for using in the early stages of design. They are specifically made for the engineering stage and therefore do not encourage experimentation. For this reason architects usually tend to use traditional media and make the shift to digital tools at a later stage. VR is traditionally used as a visualisation tool rather than a design instrument.

JCAD-VR promotes instead, the earlier use of VR thus exploiting its superior visualisation capabilities to convey a new design experience. Through JCAD-VR the user is encouraged to experiment with several design solutions done quickly. This system makes VR available to architects and at the same time it promotes their creativity.

8.3.1.1 VR-Based Conceptual Design for Architecture

JCAD-VR, albeit at a prototype level, is a tool specifically designed to support conceptual design. The interactivity of the system helps the user experiment with the design. Specifically the system has been tailored to the needs of an architect. The architect can create a number of simple shapes, such as geometric primitives, as well as architecture-related objects. In addition a library provides a number of ready-to-use 3D-objects.

Objects are created in a very simple way. The user does not need to type in values but they use mouse commands to create and change geometries.

The user can conveniently import a 3D-model of the context area and can then start designing on top of it. In this way, the user can rapidly create several design prototypes and evaluate their qualities within the real context of the design.

JCAD-VR can be considered a true VRAD system for architecture since:

- It provides real time interaction
- It hides the mathematical description of the geometries
- It encourages experimentation and creativity through a responsive visual feedback
- It helps the architect to adopt digital tools early in the design process
- It helps architects by using new technologies that can enhance the efficiency of the design process
- It promotes a user-friendly environment that does not require special expertise

8.3.1.2 User-friendly Human-Computer Interface

As shown in Chapter 6 of the companion thesis (Conti, 2002) JCAD-VR features a user-friendly Human-Computer Interface (HCI). As shown in Chapter 7 an experiment proved that typical users can become familiar with the software within a few minutes.

Users can navigate easily within the virtual world in a way that resembles a real-life experience. As shown in Section 6.5.2.1 of the companion thesis (Conti, 2002) navigation has been constrained to avoid the confusion of most VR systems. This has been done by emphasising the separation of horizontal and vertical translations and limiting rotations to those around the vertical axis.

However, interaction with the system is not only limited to navigation. As shown in Chapter 7 of the companion thesis (Conti, 2002) the user can create a number of different types of shapes through a 3D-GUI.

The adoption of the 3D-GUI brings the intuitiveness of traditional 2D-GUIs to JCAD-VR's HCI. In this way the user can control the system through an HCI that

is part of the virtual world. This is done through a number of 3D-menus and a set of tools used to communicate with other users: two 3D-panels which show images from the videoconference and the text of the chat.

For the convenience of the user, any element of the 3D-GUI can be moved, scaled or rotated just like any other object in the virtual world.

JCAD-VR proved to be easy to use even with a conventional 2D mouse. The entire system has been engineered in order to be intuitive yet effective. The user can move, rotate or scale objects in a very simple way. The user selects the object, without need for typing in values and manipulates it using a choice of 3D-widgets. This makes the interaction less error-prone, user-friendlier and very visual.

8.3.2 The Flow of Information between the Participants

As mentioned in Chapter 2 the conventional design process often limits the collaboration between groups of practitioners to an inefficient and fragmented exchange of information.

JCAD-VR instead allows a simple and reliable flow of information between the participants in the design process and provides the means for fast communication of design ideas through a number of different communication methods. In traditional offices the collaboration among the practitioners is very often still accomplished orally and through the exchange of paper based documents such as sketches and plans. In contrast JCAD-VR offers an integrated, synchronous and fully digital means for collaboration and exchange of design choices.

8.3.2.1 Design Information Exchange

JCAD-VR provides the means for establishing an effective design information exchange among remote participants and offers a major contribution towards the creation of a distributed virtual architectural office. It supports synchronous collaboration and communication of design ideas by allowing several participants to interact with virtual objects. It provides the users with a virtual design arena that encourages the collaboration of remote design teams through the possibility of direct interaction with all the virtual elements present in the shared VE.

It also enhances the communication by offering a choice of several different tools that enable the flow of design ideas among the participants.

8.3.2.2 Communication Methods

JCAD-VR provides the user with various communication methods. Its modular and flexible architecture allows the implementation of a number of communication tools that can be easily combined or used separately according to either the network bandwidth, the power of the computers or the users needs. The users can choose between the three following configurations at their convenience:

- Video, audio, chat and whiteboard
- Audio, chat and whiteboard
- Chat and whiteboard

JCAD-VR also provides the means for fully collaborative experience by the use of avatars as virtual embodiments of the participants. These communication features, along with support for the synchronous exchange of design information through a shared VE, are the most significant contributions brought by the JCAD-VR prototype to the field of architectural design.

8.3.3 Integration into a Collaborative Virtual Design Environment (CVDE) System

JCAD-VR combines features typical of both CSCW systems and VR environments. From this point of view it is an original software prototype that can be considered a successful attempt towards the integration of the VR application and the CSCW means of communication, into a comprehensive collaborative tool. In addition JCAD-VR owns all the characteristics of the new CVDEs (See Section 3.4.5) and it can be considered one of the first examples of this category of system. It supports:

- The establishment of a distributed virtual office for remote participants
- Synchronous communication and interaction

- Three-dimensional imaging and simulation
- Interactive collaboration based on virtual presence
- Ease of creative conceptualisation
- Flow of information and the sharing of knowledge
- The sharing of knowledge and expertise for virtual enterprises.

Lastly, as mentioned in Section 4.6, JCAD-VR is not a complete and fully optimised software package rather it has to be currently considered as the unpretentious proof of a concept. Its contribution then is in offering a valuable starting point towards the development of a finished commercial package.

8.3.4 Extendable Architecture and Portability

As mentioned in Section 4.4.3 JCAD-VR was developed in a modular fashion enabling the scheduling of the implementation of independent self-functioning modules and allowing concurrent software development. This open architecture gives developers the freedom to improve the system by implementing further tools or routines at any time according to their needs. From this point of view JCAD-VR inherits the qualities of the object-oriented approach that Java™ offered by providing easy extendibility, maintainability and portability to several different platforms.

8.4 Further Work

As mentioned earlier in this chapter, JCAD-VR is based on a flexible and expandable architecture that can be easily improved with newly developed tools and routines. During its development a priority list was established to set the order for the implementation of the modules of the framework.

This procedure led inevitably to the selection and implementation of some modules while others with less priority were left for further development. These lower priority modules, (marked in black in Figure 8.1), will be given in this section as examples for further development of the JCAD-VR system. Valuable suggestions for the implementation of new tools were provided by the previous experiment

described in Chapter 7. This provided a valuable occasion to collect significant feedback about the software prototype.

The following sections will outline further development of the system and it will finally describe the methodology and preparation of a cross university test that would require the involvement of a number of students from two schools of Architecture.

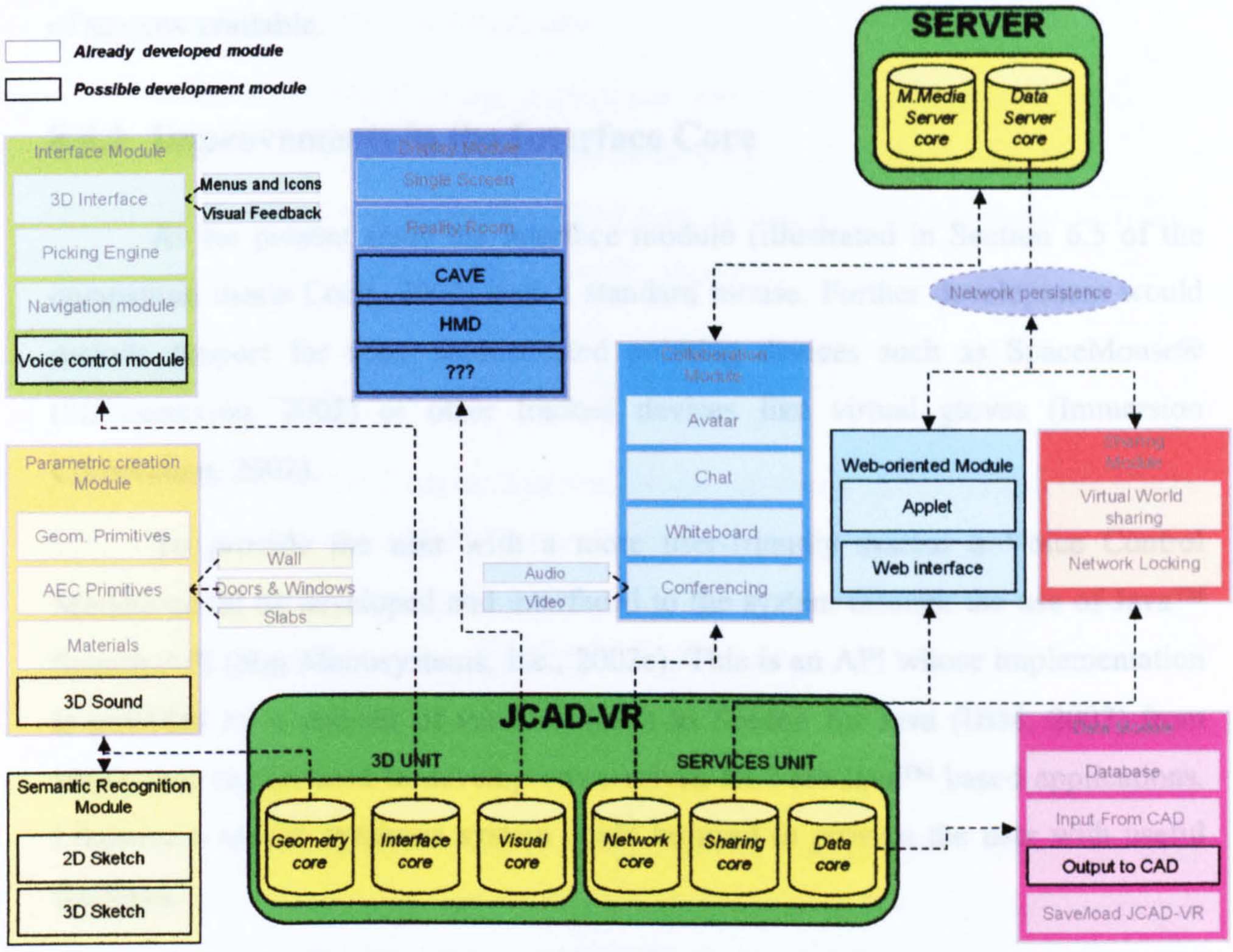


Figure 8.1: The JCAD-VR framework with the unimplemented modules marked in black

8.4.1 Enhancements to the Visual Core

Currently JCAD-VR supports only monoscopic visualisation using traditional monitors or a Reality Center™.

However, (as demonstrated in Section 6.4.3.3 of the companion thesis Conti, 2002) the display module was developed in order to be easily expandable. The

successful development of the code necessary to support stereoscopic views and tracking devices would lead to the support of more visualisation devices including CAVEs, Virtual Tables or HMD.

Additionally, support for multiple screen desktop based configurations could also be supported. In this case, the development of a "wizard" that would allow users to configure the system at start-up, could also be included. Users could configure a simple semi-immersive environment quickly, according to the number and position of screens available.

8.4.2 Improvements in the Interface Core

At the present stage the interface module (illustrated in Section 6.5 of the companion thesis Conti, 2002) uses a standard mouse. Further development would include support for more sophisticated pointing devices such as SpaceMouse® (3DConnexion, 2002) or other tracked devices like virtual gloves (Immersion Corporation, 2002).

To provide the user with a more user-friendly system a Voice Control Module could be developed and interfaced to the system through the use of Java™ Speech API (Sun Microsystems, Inc., 2002e). This is an API whose implementation is provided by a number of vendors, (such as Speech for Java (IBM, 2002) from IBM), that can be used to develop voice driven HCI for Java™ based applications. Likewise a speech synthesis system could be used to provide the user with useful feedback.

The entire Interface Core could also be re-developed to support a multimodal HCI that, based on the use of immersive visualisation technologies, would effectively combine different modalities such as sketches, gestures, speech and gaze to help the designer accomplish tasks in a natural user-centred way. In fact, as cognitive scientists have proved, the design experience strongly benefits from the support of multi-sensorial, or *multimodal*, interactions.

Different *modalities* can be considered as complementary conceptual channels that can transmit information not easily acquired spatially. One of the main advantages of the integration of different modalities lies in the widened perceptual

and conceptual bandwidth available to the user to convey information regarding the object he is creating. Such integrated approach is founded upon the effective support of human communication patterns that can provide, if combined, spatial description and mutual interrelation hardly achievable through other means.

This way JCAD-VR could be thought of as a design tool that would allow the designer to freely sketch and manipulate 3D-shapes in a transparent fashion within an immersive Virtual Environment (VE) augmented through natural interactions such as speech, gesture or gaze interpretation.

Students involved in the experiment (described in Chapter 7), proposed further improvements by the development of a panel showing a plan view of the virtual world. This would be used to provide an overview of the environment as well as to indicate to other users where a specific object is.

8.4.3 Further Developments of the Geometry Core

As illustrated in Chapter 7 of the companion thesis (Conti, 2002) the JCAD-VR Geometry Core provides the means for the creation of a number of shapes. It has been also proven how the development of fully parametric shapes can be integrated into the existing framework. Therefore, further improvements should include the development of more parametric objects. A number of existing objects, such as stairs or furniture, could also be made fully parametric.

The library of objects could be extended to include more objects and more materials. This could also include lights and pre-recorded sounds such as people chatting and street noise etc. In this way users could quickly create specific environmental conditions in the VE.

The system could also be extended by implementing Boolean functions. This could be done in Java3D™ or simply interfacing the Geometry Core with an external modelling kernel.

Another module could also provide support for semantic recognition of sketches. Similar to the Space Pen system (Jung et al., 2002, described in Section 3.4.2.5 of the companion thesis Conti, 2002) users could sketch shapes directly into

the virtual environment. However in JCAD-VR, the shape once recognised could be replaced by the relevant 3D-model, which could be sent to the other users through the network.

8.4.4 Enhancements to the Data Core

As described in Section 7.6 of the companion thesis (Conti, 2002) currently JCAD-VR loads files created with external CAD packages through a VRML97 loader (See Appendix D.5). In addition the system can also save and retrieve the content of the virtual world in jcad format. Further development would include support for more file formats. This would facilitate the integration of JCAD-VR with the existing workflow.

8.4.5 Enhancements to the Sharing Core

Enhancement to the Sharing Core would include the development of an Applet version of JCAD-VR. This would allow the system to be run inside an Internet browser. This would facilitate the use of the system since it would not require the user to install the system. This would be automatically launched once the relevant web page was loaded.

However, due to the actual size of the software (approx 50 Mb) further optimisations and a modular download system should be developed. The applet would start the download of a central core and then access the relevant sections as required. To partially solve this problem the jar archives could be cached inside the user's hard drive the first time the system is loaded.

8.4.6 Enhancements to the Collaboration Tools and Network Architecture

The network architecture of JCAD-VR could be improved by adding new collaborative tools and by optimising its transmission routines. The video conferencing system is not optimised and can result in being too demanding in terms of bandwidth usage when more than two clients are connected to the system. Under

these circumstances communication could be allowed by the transmission of only the audio channel and by replacing the video stream with a 3D representation of the participants simulating the oral communication through the movement of the mouth and facial expressions.

In the standard version, the JCAD-VR Server offers support to only one VE at a time. However, it could be modified to handle multiple worlds concurrently. At the moment this can only be achieved by launching several JCAD-VR Server applications on different computers.

The information flow about the virtual objects would be improved by optimising some of the routines for the transmission of data. For instance, at present every time the user deletes an object in the VE, the network again transmits all its parameters to the remote participants. Instead, the same information (the object being deleted) could be broadcast transmitting only the Deleted Status and the object's ID number. The same simplification could be applied in the case of the user wishing to modify of only a few of the objects parameters. This would partially reduce the amount of information that has to be transmitted across the network.

Section 6.2.2.1 described in detail the array used to transmit the parameters of virtual objects. This array contains some empty positions that could be filled with extra information about the existing objects or with parameters of newly implemented objects. The same array could be elongated to accommodate other relevant information if required.

In JCAD-VR the embodiment of a user is allowed through only one 3D model resembling a human. This is used together with 3D text showing the login names, to differentiate the participants of the session. The virtual scene could be enriched by providing the possibility of using several different 3D-models for the avatars. This would allow the possibility of choosing among embodiments resembling women or men, children, wheelchair users or simply using abstract models. For this purpose the ClientInFrame panel (described in Section 6.2.1.6) contains an empty area for the avatar tool, which could be filled with a list of models from which users could select their preferred virtual appearance.

During the test (described in Chapter 7) students suggested the implementation of a tool that would allow users to point at directions and areas in the VE while discussing design solutions. According to the students this would provide a clearer reference for the discussion. In this way users would point at an object, stating for instance that it was "North" from the square, or "East" from the red house. This issue would be solved by simply including in the virtual scene an object acting as an absolute reference for directions, e.g. a virtual compass, indicating North, South, East and West. As an alternative the system could include a routine that would allow users to point at virtual objects visually and use appropriate visual marks to annotate the object in the VE.

Students also suggested the implementation of a more sophisticated Locking Mechanism than the one included in JCAD-VR, which would still allow the possibility of choosing whether an object can be edited or not by other participants. This routine should extend the one already implemented by adding the possibility of temporarily locking objects which should not be modified by other users.

As mentioned in the introduction the methodology to run a cross-university test was developed, and therefore this is included as part of further work proposals. The following section will describe in detail the guidelines for this future experiment.

8.4.7 Preparation of the Collaborative Session between the two Universities

This section illustrates the general methodology that was developed to set up a more ambitious experiment than the in-house test described in Chapter 7. This test will involve architecture students from the University of Strathclyde in Glasgow, and from the Technische Universiteit Eindhoven (TU/e) in the Netherlands. This collaborative session has not yet been carried out due to the complexity of matching timetables and classes between the two institutions. However, the preparation work described in the following sections was included in this thesis because it offers a structured guideline for further experiments in the future.

8.4.7.1 The Methodology

The experiment would be divided into two sessions of 2 hours each, and would involve a minimum of 6 students in Glasgow and 6 students in Eindhoven (3 students at each session in both institutions).

The two sessions would be programmed as follows:

- 15 minutes to explain the exercise to the students and to distribute the material
- 1 hour and 45 minutes to fulfil the design task.

During each session 3 students in Glasgow would be working with 3 students in Eindhoven. One student at the University of Strathclyde and one at the TU/e would be matched and treated as a group. For each session there would therefore be 3 groups of 2 people.

Each group would work on the assigned task, to design an information kiosk by collaborating over the net. Different groups would use either *JCAD-VR TU/e Version* packages or emails with files as attachments. Specifically, as illustrated in Figure 8.2:

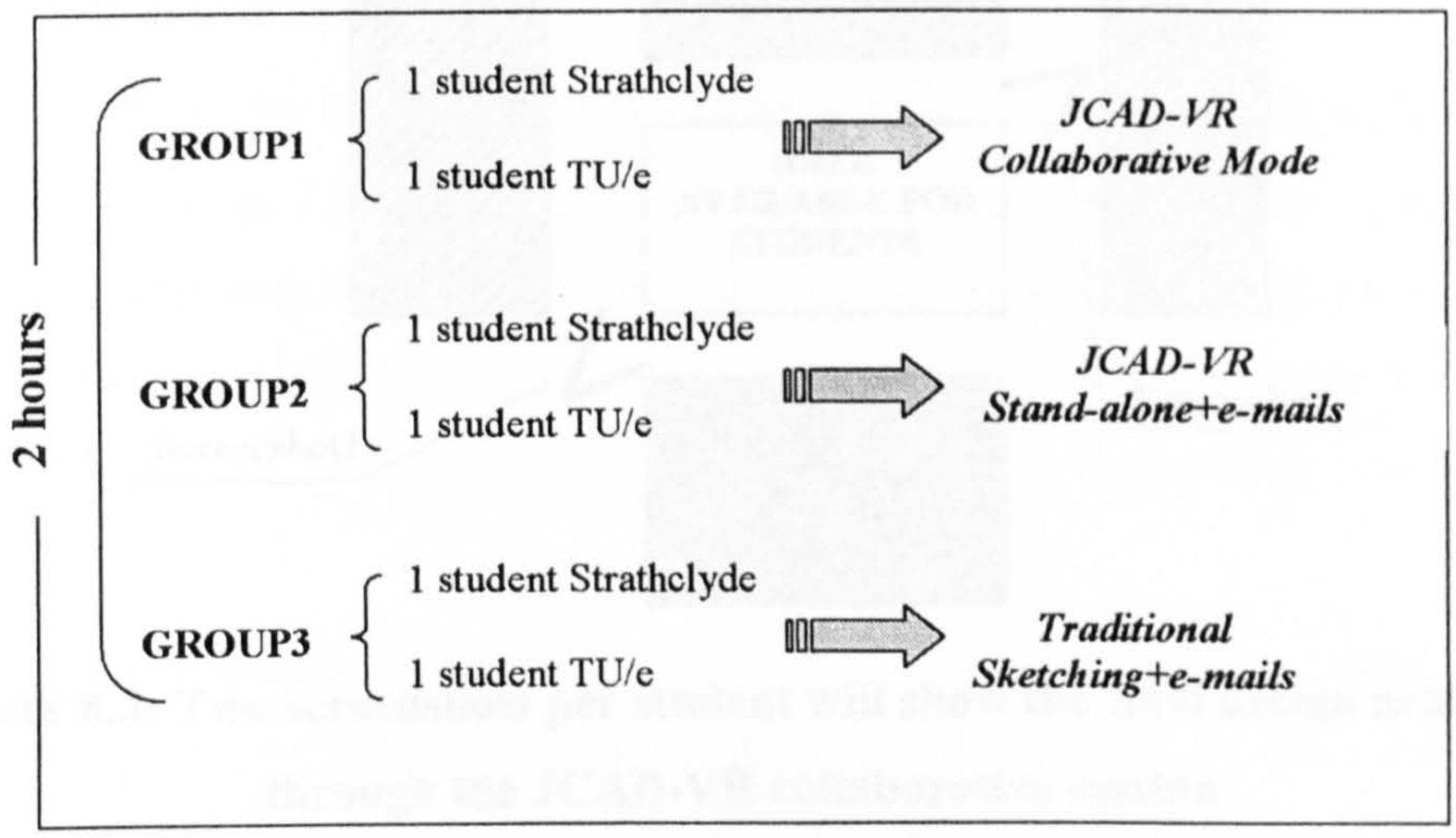


Figure 8.2: Schema for the cross-university experiment

- Group 1 would be asked to accomplish the design task during a collaborative session using JCAD-VR TU/e Version.

- Group 2 would be using JCAD-VR TU/e Version which was not connected to the network plus emails, as their means of collaboration and for sending and receiving files with the upgraded version of their design.
- Group 3 would be asked to use traditional sketches and emails to allow the flow of information.

Activity for Group 1

The couple would be asked to fulfil the design task using JCAD-VR TU/e Version running in collaborative mode. They could use JCAD-VR's tools to participate and thus they would test the capabilities of the software to let users collaborate in a synchronous manner. Both students would work concurrently on the design task as described in Appendix F.

The material to be produced at the end of the 2 hours would include 2 screenshots per student. The points of view chosen have to clearly visualize the final stage of their design. Possibly one of the two views (illustrated in Figure 8.3) should be chosen:

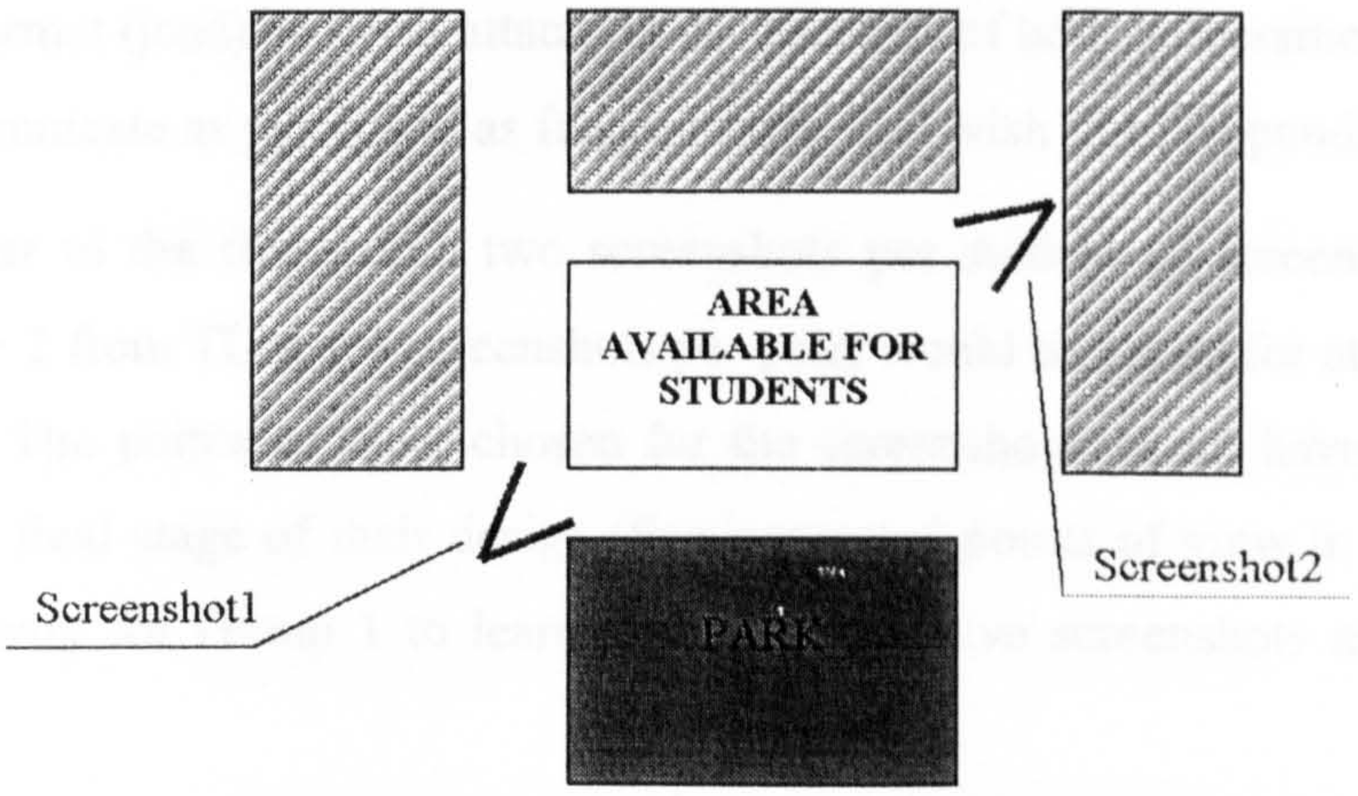


Figure 8.3: Two screenshots per student will show the final design achieved through the JCAD-VR collaborative session

These screenshots would be saved manually by clicking on the screenshot icon. JCAD-VR would then automatically assign a sequential name to each of them. These files would be eventually stored in the directory *JCAD-VR\capture*. Potentially students could take several snapshots and then choose the two most representative

ones. The final two images (in jpg format) for each student would be named *StudentName_01* and *StudentName_02* and saved in the directory *JCAD-VR\capture.*

A record track for the collaborative session of Group 1 would be kept using the automatic routines implemented in JCAD-VR TU/e Version (See Section 7.2). The system saves screenshots of the VE in jcad format automatically every 10 minutes. The entire environment would be stored with a sequential number (i.e. *screenshot 1, screenshot 2, screenshot n... File 1, File 2, File n*) in the *JCAD-VR\automatic_save* folder. In addition the content of the chat board would be recorded. These routines would help keep a sequential track of the most relevant design achievements during the experiment and would highlight any problem during the collaborative session.

Activity for Group 2

The pair would be asked to fulfil the design task described, by using JCAD-VR TU/e Version as a stand-alone application. As a consequence of not being connected to the server, the system would not activate any collaborative tools.

Students would communicate through emails and exchange files saved in JCAD-VR format (jcad) as email attachments. Students of both universities would be free to communicate as much and as frequently as they wish (See Appendix F).

Similar to the first group two screenshots per student (2 screenshots from Strathclyde + 2 from TU/e = 4 screenshots per pair) would be asked for at the end of the 2 hours. The points of view chosen for the screenshots would have to clearly visualize the final stage of their design (See suggested points of view in Figure 8.3 and see Activity for Group 1 to learn about how to save screenshots manually in JCAD-VR).

Students would be also asked to keep all the information, which flowed within their group, by keeping all the emails received and the jcad files exchanged. This would keep track of the frequency of the exchange of information and would record the evolution of the design. The emails received during the experiment would have to be saved in a separated folder whose path in the hard-drive would have to be agreed with a tutor before the beginning of the session. In addition, JCAD-VR TU/e Version (See Section 7.2) would automatically save screenshots in its internal folder

(*JCAD-VR\automatic_save*) every 10 minutes, and every 5 minutes it would perform a back-up saving of jcad files in *JCAD-VR\automatic_save*.

Activity for Group 3

The pair would be asked to fulfil the design task using traditional free hand sketching. Students would exchange scanned files of sketches etc. as email attachments and use emails to communicate. Students from both universities would be free to exchange information and files over the net as frequently as they wish. They would also require a scanner and printer in order to produce the final design. The paper format for the experiment would be A4 size.

The material to be produced at the end of the 2 hours would be, 1 perspective drawing per student (1 from Strathclyde + 1 from TU/e = 2 perspectives per group) where the point of view chosen would have to clearly visualize the final stage of their design (Figure 8.3 shows the suggested points of view).

Students would be asked to keep all the evidence of the flow of information within the group by keeping all the received emails and the exchanged files. In this way it would be possible to keep track of the frequency of the exchange of information and to record the evolution of the design. Similar to Group 2, the emails received would have to be saved in a local folder. The location of this folder in the hard-drive would have to be agreed with the tutor before the beginning of the session.

Material to be Provided to the Students to fulfill the Design Task

Once launched, the system would be ready for the experiment and supplied with the appropriate background for Group 1 and Group 2 (See Section 7.2).

For Group 3, a tutor would provide the students with several plans of the site including:

- A general plan
- An elevation of the site, showing where to locate the kiosk
- 2 sections
- 2 perspectives.

Additional Introductory Session for those Students using Software

An introductory session of one hour would be necessary to introduce the students to the software package and to allow some practice before the experiment.

This session could be arranged separately by the students in each university.

Evaluation

This experiment would help in evaluating the collaborative capabilities that JCAD-VR offers to users and it would make a comparison of it possible, against other collaborative methods. This test would allow the monitoring of the information exchange, the working conditions and the design solutions through the sequences of snapshots (for those students working with JCAD-VR TU/e Version) and the exchange of files. The consistency of all the students' final products would give the opportunity to qualitatively evaluate the efficiency of JCAD-VR against the more traditional methods. In this experiment the snapshots recorded at a fixed time interval, and the data, such as the time recorded on the emails, would provide the means for reconstructing the students' design processes. Moreover, through each group's final perspectives it would be possible to evaluate if the communication between the two members of each group was precise enough to allow them accomplish a common design task.

In addition, the tutors would be provided with a questionnaire for each of the three groups involved in the experiment (See Appendix E). Through the students answers to these questionnaires it would be possible to obtain further feedback on the effectiveness of each of the collaborative methods and about the JCAD-VR system.

8.4.7.2 The Informal Testing

The cross-university test has not taken place. However a short and informal session to check for problems in the connection between the two institutions was carried out. The main concern was on the delay in the flow of information due to the distance between the universities and on potential network problems arising from the presence of firewalls. However, this informal session was successful and proved the feasibility of a long distance experiment.

8.5 Conclusions

The JCAD-VR prototype promotes the change of usage of VR from being a mere presentation medium to being a more powerful and effective design tool. It proved the feasibility of using VR systems as the future interface for the next generation of computer aided design applications for architecture and it provided the user with multimedia tools and a network platform for enhancing communication and allowing collaboration among remote participants.

JCAD-VR is an original software package specifically developed for architects and even if not yet complete or optimised as a commercial application, it has allowed collaborative sessions and the effective exchange of design information.

This chapter summarised the contribution towards enhancement of the architectural design process brought by the JCAD-VR project, and it illustrated further developments and improvements for the prototype.

The development of JCAD-VR gave the authors the possibility to investigate and solve many theoretical and technical issues. In several occasions the authors experienced both excitement and frustration caused by the nature of the research and the fast progress of the technology involved. In particular the implementation of the Visual Core of the system took months of intense work in order to allow the use of the Virtual Environment Laboratory (VEL) (University of Strathclyde, 2002), and to achieve flexibility in the use of visual devices for the framework. Today the same issue would be solved in less time and more efficiently thanks to the newly released Java3D™ API 1.3, which supports multiple screen configurations by default through a much simpler mechanism.

This highlights the issue of working with cutting edge technology where advances in hardware and software can in some cases vanish efforts or make research results soon obsolete. However, to conclude with a personal comment the most exciting moment of the entire project was the testing of the software. During the experiment, in fact, people other than the authors were involved in using JCAD-VR and their quick understanding of the system along with their enthusiasm has been a source of great excitement and encouragement for the authors.

References

3DConnexion (2002). Product Overview - SpaceMouse® Classic. Retrieved 13th October 2002 from the World Wide Web: http://www.logicad3d.com/products/Classic.htm

Alberti, L.B. (1992). *On the Art of Building Ten Books*. The 1988 Rykwert et al. Translation. Cambridge Mass., USA: The MIT Press.

Arias, E., Eden, H., Fischer, G., Gorman, A. and Scharff, E. (2000). Transcending the Individual Human Mind - Creating Shared Understanding through Collaborative Design. *ACM Transactions on Computer-Human Interaction (TOCHI) 7(1) 2000* (pp. 84-113). Retrieved 18th June 2002 from the World Wide Web: http://doi.acm.org/10.1145/344949.345015

Atkin, B., Clarke, A. and Smith, D. (1996). *Benchmarking Best Practice Briefing and Design*. University of Salford, Construct IT Centre of Excellence.

Aukstankalnis, S. and Blatner, D. (1992). *Silicon Mirage: the Art and Science of Virtual Reality*. Berkeley, CA: Peachpit Press;

Bar, F. and Borrus, M. (1993). The Future of Networking. Berkeley, CA: University of California, BRIE working paper. Retrieved 12th June 2002 from the World Wide Web: http://brie.berkeley.edu/~briewww/pubs/rp/network.html

Baran, P. (1964). On Distributed Communications: Introduction to Distributed Communications Networks. RM-3420-PR U.S. Defence Documentation Center (DDC). Retrieved 12th June 2002 from the World Wide Web: http://www.rand.org/publications/RM/RM3420/

Bartle, R. (1990). Early MUD History. Retrieved 27th August 2002 from the World Wide Web: http://apocalypse.org/pub/u/lpb/muddex/bartle.txt

Besser, H. (1998). *The Information SuperHighway: Social and Cultural Impact.* School of Library & Information Studies, University of California, Berkeley. Retrieved 13th June 2002 from the World Wide Web: http://www.gseis.ucla.edu/~howard/Papers/brook-book.html

Bodker, S., Ehn, P., Knudsen, J., Kyng, M. and Madsen, K. (1988). Computer Support for Cooperative Design. In *Proceedings of the conference on Computer-supported cooperative work, 1988, Portland, Oregon, United States* (pp. 377-394). New York: ACM Press. Retrieved 26th June 2002 from the World Wide Web: http://doi.acm.org/10.1145/62266.62296

Bricken, W. and Coco, G. (1994). The VEOS Project. *Presence, Vol. 3, No. 2, 1994* (pp. 111-129). Retrieved 27th August 2002 from the World Wide Web: http://citeseer.nj.nec.com/bricken93veos.html

Broadbent, G. (1988). *Design in Architecture: Architecture and the Human Sciences.* London: David Fulton Publishers.

Brook, J. and Boal, I. (1995). *Resisting the Virtual Life: The Culture and Politics of Information*. San Francisco: City Lights Press.

Brown, C., Benford, S. and Snowdon, D. (1996). Collaborative Visualization of Large Scale Hypermedia Databases. In U. Busbach, D. Kerr and K. Sikkel (Eds.), *CSCW and the Web - Proceedings of the 5th ERCIM/W4G Workshop Arbeitspapiere der GMD 984, GMD, Sankt Augustin, April 1996* (pp. 115-123). Retrieved 21st August 2002 from the World Wide Web: http://orgwis.gmd.de/projects/W4G/proceedings/visual.html

Brown, S. (2001). *Communication in the Design Process.* London: Spon Press.

Burdea, G. and Coiffet, P. (1994). *Virtual Reality Technology.* USA: John Wiley & Sons.

Burka, L. (1993). A Hypertext History of Multi-User Dimensions. Retrieved 26th August 2002 from the World Wide Web: http://www.apocalypse.org/pub/u/lpb/muddex/essay/

Bylund, N., Fredricson, H. and Thompson, G. (2002). A Design Process for Complex Mechanical Structures using Property Based Models, with Application to Car Bodies. In D. Marjanović (Ed.), *Proceedings of the Design 2002, 7th International Design Conference 2002* (pp. 611-620). Zagreb, Croatia: Faculty of Mechanical Engineering and Naval Architecture, Zagreb and The Design Society, Glasgow, UK.

Capin, T.K. Thalmann, D., Pandzic , I. S. and Magnenat, N. (1998). Realistic Avatars and Autonomous Virtual Humans in VLNET Networked Virtual Environments. In J. Vince and R. Earnshaw (Eds.), *Virtual Worlds on the Internet* (pp. 157-173). Unites States: IEEE Computer Society.

Castells, M. (2000). *The Rise of the Network Society.* Great Britain: Blackwell Publishers Ltd.

Cera, C., Regli, W., Braude, I., Shapirstein, Y. and Foster, C. (2001). A Collaborative 3D Environment for Authoring of Design Semantics. *IEEE Computer Graphics and Applications, Vol. 22, No. 3, May-June 2002.* Retrieved 19th June 2002 from the World Wide Web: http://gicl.mcs.drexel.edu/MUG/

Cherry, C. (1971). *World Communication: Threat or Promise?* Great Britain: John Wiley & Sons Ltd.

Cherry, C. (1978). *On Human Communication: a Review, a Survey and a Criticism.* Cambridge, Massachusetts, U.S.A.: The Mit Press.

Cherry, C. (1985). *The Age of Access: Information Technology and Social Revolution: Posthumous Papers of Colin Cherry*. William Edmondson (Ed.) Great Britain: Croom Helm Ltd.

Christiansson, P. (1999). Experiences from Design and Use of IT supported Distributed Learning Environment. In R.M. Lloyd & C.J. Moore (Ed.), *Civil Engineering Learning Technology in Cardiff, 3rd AECEF International Symposium, 1999, Cardiff* (pp. 29-42). London: Thomas Telford Ltd.

Christiansson, P. (2000). IT in Distributed Open Learning Environments. In G. Gudnason (Ed.), *Construction Information Technology 2000- Taking the Construction Industry into the 21st century, 2000* (pp. 197-208). Reykjavik: Icelandic Building Research Institute.

Computer History Museum (2002). Retrieved 13th June 2002 from the World Wide Web: http://www.computerhistory.org/timeline/1970/index.page

Computer History Museum (2002a). Retrieved 13th June 2002 from the World Wide Web:

http://www.computerhistory.org/exhibits/internet_history/internet_history_80s.page

Conti, G. (2002). *A VR System for the Early Stages of the Design Process in Architecture*. Unpublished PhD thesis, University of Strathclyde.

Cooke, J. M., Zyda, M. J., Pratt, D. R. and McGhee, R. B. (1992). NPSNET: Flight Simulation Dynamic Modeling using Quaternions. *Presence: Teleoperators and Virtual Environments, Vol. 1, No 4, Fall 1992* (pp. 404-420). Retrieved 27th August 2002 from the World Wide Web: http://www.movesinstitute.org/~zyda/pubs/Presence.1.4.pdf

COVEN (2002). Retrieved 27th August 2002 from the World Wide Web: http://coven.lancs.ac.uk/home.htm

COVEN (2002a). Retrieved 27th August 2002 from the World Wide Web: http://chinon.thomson-csf.fr/projects/coven/PICTURES/demos/game.jpg

COVEN (2002b). Retrieved 27th August 2002 from the World Wide Web: http://chinon.thomson-csf.fr/projects/coven/PICTURES/demos/office.jpg

COVEN (2002c). Retrieved 27th August 2002 from the World Wide Web: http://chinon.thomson-csf.fr/projects/coven/PICTURES/demos/ucl.jpg

COVEN (2002d). Retrieved 27th August 2002 from the World Wide Web: http://www.iis.gr/main/projects/COVEN/VTA.htm

Crabtree, A., Pettifer, S. and Rouncefield, M. (2000). Designing Virtual Environments for Social Interaction. In *Proceedings of Virtual Reality International Conferences 2000* (pp. 118-129). Laval, France: VR News. Retrieved 27th August 2002 from the World Wide Web: http://www.mrl.nott.ac.uk/~axc/documents/Laval_Virtual_2000.pdf

Cruz-Neira, C., Sandin, D., DeFanti, T.A. (1993). Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE. In *Proceedings of SIGGRAPH 93 Computer Graphics Conference, 1993, ACM SIGGRAPH* (pp. 135-142). Retrieved 15th August 2002 from the World Wide Web: http://www.evl.uic.edu/EVL/RESEARCH/PAPERS/CRUZ/sig93.paper.html

Cybertimes (2000). Did you know....VR Engineering? Posted November 2000. Retrieved 02nd December 2001 from the World Wide Web: http://www.cyberminduk.com/newsletter/cybertimes7.htm

Dagit, C.E.III (1993). Establishing Virtual Design Environments in Architecture Practice. In U. Flemming and S. Van Wyk (Eds.), *CAAD Futures '93 Proceedings of the fifth international conference on Computer-Aided Architectural Design futures 1993* (pp. 513-521). North-Holland: Elsevier Science Publishers B.V.

Darken, R. and Zyda, M. (2001). *VR - The Historical Context.* Unpublished presentation. Retrieved 23rd August 2002 from the World Wide Web: http://www.npsnet.org/~zyda/courses/1HistoryOfVR.pdf

Delargy, M. (1999). A New Dimension. *Building 02 July 1999.* Retrieved 17th June 2002 from the World Wide Web: http://www.building.co.uk/story.asp?sectioncode=71&storyCode=5708

DIME (2002). Retrieved 27th August 2002 from the World Wide Web: http://www.sics.se/~olovs/dime/

Distributed Legible City (2002). Retrieved 17th August 2002 from the World Wide Web: http://aig.cs.man.ac.uk/gallery/legible.html

DIVE (2002). Retrieved 27th August 2002 from the World Wide Web: http://www.sics.se/dce/dive/dive.html

Do, E. (2001). Sketching Interfaces for Conceptual Design and Analysis in Architecture. In *CHI Workshop, Tools, Conceptual Frameworks and Empirical Studies for Early Stages of Design, April 2001, Seattle, U.S.* Retrieved 19th June 2002 from the World Wide Web: http://depts.washington.edu/dmgwksp/PP/do.pdf

Dorta, T. and LaLande, P. (1998). The Impact of Virtual Reality on the Design Process. In S. van Wyk and T. Seebohm (Eds.), *Digital Design Studios: Do Computers Make a Difference in Design Studio? ACADIA Conference Proceedings 1998, Québec City, Québec, Canada, 22-25 October* (pp. 138-160).

Dourish, P. and Bellotti, V. (1992). Awareness and Collaboration in Shared Workspaces. In *Proceedings of the conference on Computer-supported cooperative work, Toronto, Ontario, Canada, 1992* (pp. 107-114). New York: ACM Press. Retrieved 28th August 2002 from the World Wide Web: http://doi.acm.org/10.1145/143457.143468

Dutton, W.H. (2001). *Society on the line.* New York: Oxford University Press Inc.

Eckel, B. (2000). *Thinking in Java, 2nd Edition.* Retrieved 07th August 2002 from the World Wide Web: http://www.mindview.net/Books/TIJ/

Ennis, G. and Maver, T. (2001). Visit VR Glasgow-Welcoming Multiple Visitors to the Virtual City. In Hannu Penttilä (Ed.), *Proceedings of the 19th Conference on Education in Computer Aided Architectural Design in Europe, Helsinki, 2001* (pp. 423-429). Espoo, Finland: Otamedia Oy.

eRENA (2002). Retrieved 27th August 2002 from the World Wide Web: http://www.crg.cs.nott.ac.uk/research/projects/eRENA/

Evans, C. (1971). Chatting with Computers. In N. Cross (Ed.), *Proceedings of the Design Research Society, International Conference Sept 1971 on 'Design Participation' 1971* (pp. 68-72). London: Academy Editions.

Feather, J. (1998). *The Information Society: a Study of Continuity and Change.* London: Library Association Publishing.

Fischer, G. (1999). Symmetry of Ignorance, Social Creativity, and Meta-Design. In *Proceedings of the third conference on Creativity & cognition, Loughborough, United Kingdom, 1999* (pp. 116-123). New York: ACM Press. Retrieved 24th June 2002 from the World Wide Web: http://doi.acm.org/10.1145/317561.317582

Fisher, E., Haines, R. and Price, T. (1980). *Cognitive Issues in Head-Up Displays,* NASA Technical Paper 711. Washington, D.C.: NASA

Foster, C. V., Shapirshteyn, Y., Cera, C. D. and Regli, W. C. (2001). Multi-User Modeling of NURBS-Based Objects. In *Proceedings of DETC'01, ASME Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Sept. 2001. Pittsburgh, Pennsylvania.* Retrieved 28th

August 2002 from the World Wide Web: http://edge.mcs.drexel.edu/GICL/papers/PDFs/ASME-DETC2001-CIE-21256.pdf

Frécon, E. and Stenius, M. (1998). DIVE: A Scaleable Network Architecture for Distributed Virtual Environments. *Distributed Systems Engineering Journal (DSEJ) Vol. 5, No. 3, Sept. 1998, Special Issue on Distributed Virtual Environments* (pp. 91-100). The Institution of Electrical Engineers, Institute of Physics Publishing, The British Computer Society. Retrieved 21st August 2002 from the World Wide Web: http://www.sics.se/~emmanuel/publications/dsej/dsej.html

Fujimoto, R. (2002). *Parallel and Distributed Simulation Systems*. Retrieved 14th August 2002 from the World Wide Web: http://www.cc.gatech.edu/classes/AY2002/cs7001_fall/Notes/fujimoto.ppt

Geisler, C. and Rogers, E.H. (2000). Technological Mediation for Design Collaboration. In *Proceedings of IEEE professional communication society international professional communication conference and ACM special interest group on documentation conference on Technology & teamwork, Cambridge, Massachusetts, 2000* (pp. 395-405). Piscataway, NJ, USA: IEEE Educational Activities Department.

Ghee, S. (1995). dVS : a Distributed VR Systems Infrastructure. In A. Lastra and H. Fuchs (Eds.), *Course Notes: Programming Virtual Worlds, SIGGRAPH '95, 1995* (pp. 6-1, 6-30).

Gibson, W. (1989). *Neuromancer.* New York: Grafton.

Greenhalgh, C. and Benford, S. (1995). MASSIVE: a Collaborative Virtual Environment for Teleconferencing. *ACM Transactions on Computer-Human Interaction, Vol. 2, No. 3, September 1995* (pp. 239-261). New York: ACM Press. Retrieved 17th August 2002 from the World Wide Web: http://doi.acm.org/10.1145/210079.210088

Gribble, C. (2001). *History of the Web beginning at CERN.* Posted October 2001. Retrieved 10th June 2002 from the World Wide Web: http://www.hitmill.com/internet/web_history.html#mosaic

Grudin, J. (1994). CSCW: History and Focus. *IEEE Computer 27(5) May 1994*,19-27. Retrieved 25th June 2002 from the World Wide Web: http://www.ics.uci.edu/~grudin/Papers/IEEE94/IEEEComplastsub.html

Hansen, P.K., Gubi, E. and Hansen, G. (2002). Modularity and Distributed Product Development. In D. Marjanović (Ed.), *Proceedings of the Design 2002, 7th International Design Conference 2002* (pp. 641-646). Zagreb, Croatia: Faculty of Mechanical Engineering and Naval Architecture, Zagreb and The Design Society, Glasgow, UK.

Havelock, E. A. (1982). *The Literate revolution in Greece and its Cultural Consequences*. Princeton, NJ: Princeton University Press.

Hirschberg, U., Gramazio, F., Höger, K., Legendre, G.L., Milano, M. and Stäger, B. (2000). EventSpaces: A Multi-Author game and Design Environment. In Dirk Donath (Ed.), *Proceeding of the 18th Conference on Education in Computer Aided Architectural Design in Europe, 2000* (pp. 65-72). Bauhaus Universität, Weimar, Germany.

HIVE (2002). Retrieved 25th August 2002 from the World Wide Web: http://www.crg.cs.nott.ac.uk/research/projects/HIVE/

Horváth, I. and Rusák, Z. (2001). Collaborative Shape Conceptualization in Virtual Design Environments. *Communications of the ACM, Vol. 44, No. 12, December 2001* (pp. 59-63). New York: ACM Press. Retrieved 28th August 2002 from the World Wide Web: http://doi.acm.org/10.1145/501338.501345

Horváth, I., Vergeest, J. S. M. and Rusák, Z. (2002). Six ingredients of Collaborative Virtual Design Environments. In D. Marjanovic (Ed.), *Proceedings of the Design 2002, 7th International Design Conference 2002* (pp. 67-74). Zagreb, Croatia: Faculty of Mechanical Engineering and Naval Architecture, Zagreb and The Design Society, Glasgow, UK.

Hughes, M., Hughes, C., Shoffner, M. and Winslow, M. (1997). *Java Network Programming*. Greenwich, CT, USA: Manning Publications Co.

IBM (2002). Speech for Java. Retrieved 13th October 2002 from the Word Wide Web: http://www.alphaworks.ibm.com/tech/speech

Immersion Corporation (2002). Hardware Products. Retrieved 13th October 2002 from the World Wide Web: http://www.immersion.com/products/3d/interaction/hardware.shtml

Internet Society (2002), Retrieved 18th June 2002 from the World Wide Web: http://www.isoc.org/internet/history/brief.shtml

Ito, Y. (1991). Johoka as a Driving Force of Social Change. *Keio Communication Review 12* (pp. 33-58).

JMF™ (2002). JMF API Guide. Retrieved 11th September 2002 from the World Wide Web: http://java.sun.com/products/java-media/jmf/2.1.1/guide/

Johnson, A., Roussos, M., Leigh, J., Barnes, C., Vasilakis, C., Moher, T. (1998). The NICE Project: Learning Together in a Virtual World. In *Proceedings of VRAIS '98, Atlanta, Georgia, March 14-18, 1998* (pp. 176-183). Retrieved 26th August 2002 from the World Wide Web: http://www.evl.uic.edu/tile/NICE/NICE/PAPERS/VRAIS/vrais98.2.html

Johnson, A. and Leigh, J. (2001). Tele-immersive Collaboration in the CAVE Research Network. In Elizabeth F. Churchill, David N. Snowdon and Alan J. Munro (Eds.), *Collaborative Virtual Environments: Digital Places and Spaces for Interaction, 2001* (pp. 225-243). London: Springer.

Johnson, B.R. (2000a). Sustaining Studio Culture: How Well do Internet Meet the Needs of Virtual Design Studios? In Dirk Donath (Ed.), *Proceeding of the 18th Conference on Education in Computer Aided Architectural Design in Europe, 2000* (pp. 15-21). Bauhaus Universität, Weimar, Germany.

Johnson, B.R. (2000b). Between Friends: Support of Workgroup Communications. In Mark Clayton and Guillermo Vasquez de Velasco (Eds.), *Eternity, Infinity and Virtuality in Architecture, Proceedings of the 22nd Annual Conference of*

the Association for Computer-Aided Design in Architecture, Washington, D.C. USA (pp. 1-14)

Jun, H.J. and Gero, J.S. (1997). Representation, Re-representation and Emergence in Collaborative Computer-Aided Design. In M.L. Maher, J.S. Gero and F. Sudweeks (Eds.), *Preprints formal aspects of collaborative computer-aided design, Key Centre of Design Computing, University of Sidney, Sydney* (pp. 303-320)

Jung, T., Gross, M. D. and Do, E. Y. (2002). Annotating and Sketching on 3D Web Models. In *Proceedings of the 7th International Conference on Intelligent User Interfaces, San Francisco, California, USA* (pp. 95-102). New York: ACM Press. Retrieved 14th October 2002 from the World Wide Web: http://doi.acm.org/10.1145/502716.502733

Kirsner, S. (1998). The Legend of Bob Metcalfe. *Wired 6/11 November 1998.* Retrieved 27th June 2002 from the World Wide Web: http://www.wired.com/wired/archive/6.11/metcalfe_pr.html

Kramarae, C. (1995). A backstage Critique of Virtual Reality. In S.G. Jones (Ed.), *CyberSociety: Computer-mediated communication and community 1995* (pp. 36-56). Thousand Oaks, CA, US: SAGE Publications.

Kvan, T. (1997). But is it Collaboration? In B. Martens, H. Linzer and A. Voigt (Eds.), *Proceedings of 15th eCAADe conference, 1997* (pp. 48-55). Vienna, University of Technology. Retrieved 20th June 2002 from the World Wide Web: http://arch.hku.hk/people/tkvan/ecaade97.html

Laiserin, J. (1999a). The Future of AEC Technology. *CADENCE-1/99.* Retrieved 26th June 2002 from the World Wide Web: http://www.cadenceweb.com/1999/0199/issuefocus0199.html

Laiserin, J. (1999b). A New Kind of CAD-Communication-Aided Design. *CADENCE-6/99.* Retrieved 26th June 2002 from the World Wide Web: http://www.cadenceweb.com/1999/0699/issuefocus0699.html

Laiserin, J. (1999c). Web-Based Project Management Software: Internet-based Project Collaboration allows Far-Flung Team Members to Design Buildings

Without Flying around to Meetings. *Architectural Record-9/99.* Retrieved 27th June 2002 from the World Wide Web: http://archrecord.construction.com/DIGITAL/DA_ARTIC/DA9_99.ASP

Laiserin, J. (2000a). Collaborative AEC Design. *CADENCE-11/00.* Retrieved 26th June 2002 from the World Wide Web: http://www.cadenceweb.com/2000/1100/issuefocus1100.html

Laiserin, J. (2000b). Cultural Revolution. *CADENCE-11/00.* Retrieved 27th June 2002 from the World Wide Web: http://www.cadenceweb.com/2000/1100/collaboration1100.html

Laiserin, J. (2000c). The Pre-History or Internet Collaboration. *CADENCE-12/00.* Retrieved 28th June 2002 from the World Wide Web: http://www.cadenceweb.com/2000/1200/collaboration1200.html

Lawson, B. (1990). *How Designers Think.* London, Boston: Butterworth Architecture.

Leigh, J., Barnes, C. and Vasilakis, C.A. (1995). CASA: Computer Augmentation for Smart Architectonics. Retrieved 15th August 2002 from the World Wide Web: http://www.evl.uic.edu/spiff/casa

Leigh, J., Johnson, A.E., Vasilakis, C.A. and DeFanti, T.A. (1996). Multi-perspective Collaborative Design in Persistent Networked Virtual Environments. In *Proceedings of the IEEE Virtual Reality Annual International Symposium (VRAI'96), 1996* (pp. 253-260, 271-272). Santa Clara, CA, U.S.A.. Retrieved 15th August 2002 from the World Wide Web: http://www.evl.uic.edu/EVL/RESEARCH/PAPERS/LEIGH/leigh2.html

Leigh, J., Johnson, A.E., DeFanti, T.A. (1997). CAVERN: A Distributed Architecture for Supporting Scalable Persistence and Interoperability in Collaborative Virtual Environments. *Virtual Reality: Research, Development and Applications, vol 2, no 2* (pp. 217-237). Retrieved 15th August 2002 from the World Wide Web: http://www.evl.uic.edu/cavern/cavernpapers/vrs/index.html

Leiner, B., Cerf, V., Clark, D., Kahn, R., Kleinrock, L., Lynch, D., Postel, J., Roberts, L. and Wolff, S. (2000). A Brief History of the Internet. Internet Society. Retrieved 13th June 2002 from the World Wide Web: http://www.isoc.org/internet/history/brief.shtml#PB64

Lindemann, J. (1996). *Low Cost Distributed VR.* KBS-Media Lab, Master Thesis, Lund University. Retrieved 12th August 2002 from the World Wide Web: http://it.civil.auc.dk/it/reports/vrlindemann/lcdvr.html

Lucent Technologies, Bell Labs Innovations (2002). The Evolution of Packet Switching. Retrieved 10th June 2002 from the World Wide Web: http://www.bell-labs.com/technology/packet/evolution.html

Macedonia, M. R., Zyda, M. J., Pratt, D. R., Barham, P. T. and Zeswitz, S. (1994). NPSNET: a Network Software Architecture for Large Scale Virtual Environments. *Presence Vol. 3, No 4, 1994* (pp. 265-287). Retrieved 22nd August 2002 from the World Wide Web: http://citeseer.nj.nec.com/macedonia94npsnet.html

Macedonia, M. R., Brutzman, D. P., Zyda, M. J., Pratt, D. R., Barham, P. T., Falby, J. and Locke, J. (1995). NPSNET: a Multi-Player 3D Virtual Environment over the Internet. In *Proceedings of the 1995 symposium on Interactive 3D graphics, 1995, Monterey, California, United States* (pp. 93, 94 and 210). New York: ACM Press. Retrieved 22nd August 2002 from the World Wide Web: http://doi.acm.org/10.1145/199404.199419

Marchand, M. (2001). E-Mail, Internet Pioneer at Verizon's BBN Technologies Earns Webby Award for Lifetime Achievement. BBN Technologies. Retrieved 13th June 2002 from the World Wide Web: http://www.bbn.com/news/pr-010906a.html

MASSIVE (2002). Retrieved 26th August 2002 from the World Wide Web: http://www.crg.cs.nott.ac.uk/events/Now96/arena.html

Maver, T., Petric, J., Ennis, G. and Lindsay, M. (2000). Visiting the Virtual City. In José Ripper Kós, Adriana Simeone and Rodrigo Cury Paraizo (Eds.),

Proceedings of the 4th SiGraDI'2000, Rio de Janeiro, (pp. 135-139). Rio de Janeiro: SIGraDi - Sociedade Iberoamericana de Gráfica Digital

McCarty, W.D., Sheasby, S., Amburn, P., Stytz, M.R. and Switzer, C. (1994). A Virtual Cockpit for a Distributed Interactive Simulation. *IEEE Computer Graphics and Applications, 14(1)* (pp. 49-54). United States: IEEE Computer Society.

Metcalfe, R. (1996). Remarks of Dr. Robert Metcalfe at the University of Virginia May 30, 1996. Retrieved 26th June 2002 from the World Wide Web: http://americanhistory.si.edu/csr/comphist/montic/metcalfe.htm#me7

Microsoft Corp. (2002). Microsoft DirectX®, Technology Overview. Retrieved 09th September 2002 from the Word Wide Web: http://www.microsoft.com/windows/directx/productinfo/overview/default.asp

Modin, J. (1995). COOCOM: New ways of using Information Technology for Building Design and Management. Retrieved 20th June 2002 from the World Wide Web: http://it.civil.auc.dk/it/reports/coocom1_6_1995.pdf

Morningstar, C. and Farmer, F. R. (1990). The Lessons of Lucasfilm's Habitat. In Michael Benedikt (Ed.), *Cyberspace: First Steps* (pp. 273-302). Cambridge, Massachusetts: MIT press. Retrieved 23rd August 2002 from the World Wide Web: http://www.ibiblio.org/dbarberi/papers/vcomm/

MUG (2002). Retrieved 26th August 2002 from the World Wide Web: http://gicl.mcs.drexel.edu/MUG/

Multimedia Networking for Inhabited Television (2002). Retrieved 26th August 2002 from the World Wide Web: http://www.crg.cs.nott.ac.uk/research/projects/mnit/

Neuman, R. (1991). *The Future of Mass Audience.* New York: Cambridge University Press.

NICE (2002). Retrieved 28th August 2002 from the World Wide Web: http://www.evl.uic.edu/tile/NICE/NICE/intro.html

NICE (2002a). Retrieved 28th August 2002 from the World Wide Web: http://www.evl.uic.edu/tile/NICE/NICE/somescenes.html

Nickerson, R. (1997). A Taxonomy of Collaborative Applications. In *Proceedings of the AIS 1997 Americas Conference on Information Systems, 1997* (pp. 560-562). Retrieved 28th June 2002 from the World Wide Web: http://hsb.baylor.edu/ramsower/ais.ac.97/papers/nickers.htm

Normand, V. and Tromp, J. (1996). Collaborative Virtual Environments: the COVEN Project. In *Proceedings of the Framework for Immersive Virtual Environments, FIVE'96, December 1996, Pisa.* Retrieved 15th August 2002 from the World Wide Web: http://coven.lancs.ac.uk/4/proceedings/five96.html

Normand, V. (1998). COVEN project thomson-csf page. Retrieved 15th August 2002 from the World Wide Web: http://chinon.thomson-csf.fr/projects/coven/overview.html

NPSNET-V (2002). Retrieved 15th August 2002 from the World Wide Web: http://www.npsnet.org/~npsnet/v/index.html

OpenGL (2002). What is OpenGL, why you need it and where you get it. Retrieved 08th September 2002 from the Word Wide Web: http://www.opengl.org/users/about/index.html

Orlikowski, W.J. (1992). Learning from Notes: Organizational Issues in Groupware Implementation. In *Proceedings of the conference on Computer-supported cooperative work, 1992, Toronto, Ontario, Canada* (pp. 362-369). New York: ACM Press. Retrieved 27th June 2002 from the World Wide Web: http://doi.acm.org/10.1145/143457.143549

Pandzic, I., Çapin, T., Thalmann, N. and Thalmann, D. (1995). VLNET: A Networked Multimedia 3D Environment with Virtual Humans. In *Proceedings of the Multi-Media Modeling MMM '95 Conference, Singapore, Nov. 1995* (pp. 21-32). Retrieved 13th August 2002 from the World Wide Web: http://ligwww.epfl.ch/~thalmann/papers.dir/MMM95.VLNET.pdf

Peng, C., Chang, D. C., Jones, P. B. and Lawson, B. (2002). Exploring Urban History and Space Online: Design of the Virtual Sheffield Application. *Design*

Studies, Vol. 23, No 5, Sept. 2002 (pp. 437-453). Great Britain: Elsevier Science Ltd.

Pennington, D.C (1986). *Essential Social Psychology*. London: Arnold.

Pettifer, S., West, A., Murray, C., and Crabtree, A. (1999). Designing Shared Virtual Environments for Social Interaction. In *Proceedings of the Annual Conference of the European Network for Intelligent Information Interfaces (Community of the Future)* (pp. 53-56). Siena, Italy: Division of Informatics (Edinburgh University). Retrieved 27th August 2002 from the World Wide Web: http://citeseer.nj.nec.com/251250.html

Pfeifer, J., Koo, K., Yin, P. (1995). *What is CSCW and Groupware?* Retrieved 25th June 2002 from the World Wide Web: http://ksi.cpsc.ucalgary.ca/courses/547-95/pfeifer/cscw_definition.html

Piller, C. (1994). Dreamnet: Consumers want more than TV overload from the Information Superhighway. But will they get it? *MacWorld*. Retrieved 13th June 2002 from the World Wide Web: http://www.macworld.com/1994/10/features/1141.html

Pimentel, K. and Teixeira, K. (1993). *Virtual Reality: Through the New Looking Glass*. New York: Windcrest McGraw-Hill.

Pimentel, K. and Blau, B. (1994). Teaching Your System to Share. *IEEE Computer Graphics and Applications, 14(1)* (pp. 60-65). United States: IEEE Computer Society.

Plan Newsletter (1998). Dreaming and Designing and Developing at a Distance: The Growing Use of Computers and Virtual Teamwork in Architecture and Planning. School of Architecture and Planning MIT, Plan 49. Retrieved 20th June 2002 from the World Wide Web: http://loohooloo.mit.edu/plan/plan_issues/49/dreaming/article_bottom.html

Poltrock, S. and Grudin, J. (1994). Computer Supported Cooperative Work and Groupware. In *Proceedings of the CHI'94 conference companion on Human factors in computing systems 1994, Boston, Massachusetts, USA* (pp. 355-356).

New York: ACM Press. Retrieved 27th June 2002 from the World Wide Web: http://doi.acm.org/10.1145/259963.260448

Postman, N. (1985). Amusing Ourselves to Death: Public Discourse in the Age of Show Business. New York: Penguin Books.

Qun, L., Tay, F.E.H., Cheng, T.K., Kuen, K.K.and Feng, Z.Y. (2000). CyberCAD: Internet Distributed Interactive Collaborative Design. In *Proceedings of IGRID 2000, University of Illinois, Chicago, 2000*. Retrieved 21st June 2002 from the World Wide Web: http://eicu.tp.edu.sg/publications/gdm/cybercad%20iws2000.Pdf

Ragusa, J. M. and Bochenek, G. M. (2001). Collaborative Virtual Design Environments: Introduction. *Communications of the ACM, Vol. 44, No. 12, December 2001* (pp. 40-43). Retrieved 27th August 2002 from the World Wide Web: http://doi.acm.org/10.1145/501338.501339

Reid, E. (1995). Virtual Worlds: Culture and Imagination. In S.G. Jones (Ed.), *CyberSociety: Computer-mediated communication and community, 1995* (pp. 164-183). Thousand Oaks, CA, US: SAGE Publications.

Reimann, M. and Huq, F. (1992). A Comparative Analysis Approach for Evaluating the Effects that Concurrent Engineering has on Product Life Cycle Cost. In *Proceedings of the Second International Conference on Flexible Automation and Information Management, 1992* (pp. 248-260). Washington, D.C.. Retrieved 20th June 2002 from the World Wide Web: http://www.uta.edu/infosys/reimann/acrodocs/c001.pdf

RTP (2002). RTP: A Transport Protocol for Real-Time Applications (RFC 1889). Retrieved 13th September 2002 from the World Wide Web: http://www.ietf.org/rfc/rfc1889.txt

Rutkowski, A. M. (1991). *Mission Report on INET'91*. ITU, Geneva. Retrieved 13th June 2002 from the World Wide Web: http://www.wia.org/ISOC/itu_mission.htm

Schneider, E. (1996). *alt.cyberpunk Frequently Asked Questions list.* Retrieved 29th August 2002 from the World Wide Web: http://www.faqs.org/faqs/cyberpunk-faq/

Sener, B., Vergeest, J. S. M. and Akar, E. (2002). New Generation Computer-Aided Design Tools: Two Related Research Projects Investigating the Future Expectations of Designers. In Marjanovic (Ed.), *Proceedings of the Design 2002 7th International Design Conference, Dubrovnik, Croatia, Vol. 1* (pp. 539-544). Glasgow: The Design Society.

Shaffer, A.B. (1995). *Integration Of Hypermedia Capability Into Npsnet IV.8, A Large-Scale Real-Time Distributed Simulation System*. Unpublished Master Thesis, Naval Postgraduate School Monterey. Retrieved 22nd August 2002 from the World Wide Web: http://citeseer.nj.nec.com/shaffer95integration.html

Shakarchi, A.Y. (2000). Tools for Distributed Design Practice. In Dirk Donath (Ed.), *Proceeding of the 18th Conference on Education in Computer Aided Architectural Design in Europe, 2000* (pp. 89-92). Bauhaus Universität, Weimar, Germany.

Shukur, F. (2000). *Lost Information in Design*. Lund, Sweden: KF-Sigma.

Snowdon, D. and West, A. (1994). AVIARY: Design Issues for Future Large-Scale Virtual Environments. *Presence, Vol. 3, No. 4, 1994* (pp. 288-308). Retrieved 28th August 2002 from the World Wide Web: http://citeseer.nj.nec.com/snowdon94aviary.html

Snowdon, D., Churchill, E. F. and Munro, A. (2001). Collaborative Virtual Environments: Digital Spaces and Places for CSCW: An Introduction. In Elizabeth F. Churchill, David N. Snowdon and Alan J. Munro (Eds.), *Collaborative Virtual Environments: Digital Places and Spaces for Interaction, 2001* (pp. 3-17). London: Springer.

Sowizral, H. A. and Deering, M. F. (1999). The Java 3D™ API and Virtual Reality. *IEEE Computer Graphics and Applications, May/June 1999, Vol. 19, N. 3* (pp. 12-15).

Stefik, M., Bobrow, D. G., Foster, G., Lanning, S. and Tatar, D. (1987). WYSIWIS revised: Early Experiences with Multiuser Interfaces. *ACM Transactions on Information Systems (TOIS), Vol. 5, No. 2, 1987* (pp. 147-167). New York: ACM Press. Retrieved 28th August 2002 from the World Wide Web: http://doi.acm.org/10.1145/27636.28056

Stenius, M. (1996). *Collaborative Object Modelling in Virtual Environments.* Unpublished Master Thesis. School of Computer Science and Engineering, Royal Institute of Technology, KTH, Stockholm, Sweden. Retrieved 21st August 2002 from the World Wide Web: http://www.student.nada.kth.se/~d90-mst/masters/

Stix, G. (2001). Getting More from Moore's. *Scientific American.com April 2001.* Retrieved 26th June 2002 from the World Wide Web: http://www.sciam.com/article.cfm?articleID=00060893-20F0-1C70-84A9809EC588EF21&pageNumber=1&catID=2

Suler, J. (2000). Extending a Work Group into Cyberspace. In *The Psychology of Cyberspace* (orig. pub. 1996). Retrieved 05th June 2002 from the World Wide Web: http://www.rider.edu/users/suler/psycyber/extendgrp.html

Suler, J. (2002). The Basic Psychological Features of Cyberspace. In *The Psychology of Cyberspace* (orig. pub. 1996). Retrieved 05th June 2002 from the World Wide Web: http://www.rider.edu/users/suler/psycyber/basicfeat.html

Sun Microsystems, Inc. (1998). The Java 3D™ API: for Developers and End-Users. Retrieved 06th September 2002 from the Word Wide Web: http://java.sun.com/products/java-media/3D/collateral/presentation/

Sun Microsystems, Inc. (2000). Java 3D™ API Datasheet. Retrieved 06th September 2002 from the Word Wide Web: http://java.sun.com/products/java-media/3D/collateral/java3d.pdf

Sun Microsystems, Inc. (2002a). What is Java™ Technology ?. Retrieved 08th September 2002 from the Word Wide Web: http://java.sun.com/java2/whatis.htm

Sun Microsystems, Inc. (2002b). Java 3D™ API. Retrieved 06th September 2002 from the Word Wide Web: http://java.sun.com/products/java-media/3D/

Sun Microsystems, Inc. (2002c). Release Notes Java™ 2 SDK, Standard Edition Version 1.3.1. Retrieved 06th September 2002 from the Word Wide Web: http://java.sun.com/j2se/1.3/relnotes.html

Sun Microsystems, Inc. (2002d). Java™ Media Framework API. Retrieved 06th September 2002 from the Word Wide Web: http://java.sun.com/products/java-media/jmf/index.html

Sun Microsystems, Inc. (2002e). Java™ Speech API. Retrieved 13th October 2002 from the Word Wide Web: http://java.sun.com/products/java-media/speech/

Suwa, M., Gero, J. S. and Purcell, T. (1998). The roles of Sketches in Early Conceptual Design Processes. In L. Erlbaum (Ed.), *Proceedings of Twentieth Annual Meeting of the Cognitive Science Society, 1998* (pp. 1043-1048). Hillsdale, New Jersey, US. Retrieved 20th June 2002 from the World Wide Web: http://www.arch.su.edu.au/~john/publications/PDF2/98SuwaGeroPurcellCogSci.html

Telleen, S. (1995). IntraNet Methodology: Concepts and Rationale (white paper). Retrieved 26th June 2002 from the World Wide Web: http://www.iorg.com/papers/amdahl/concepts1.html

The Glasgow Directory (2002). Retrieved 28th August 2002 from the World Wide Web: http://iris.abacus.strath.ac.uk/gary/Miscellaneous/final/

The new Collins Concise English Dictionary (1982). William T. McLeod (Ed.). Glasgow: William Collins Sons & Co. Ltd.

Turner, A., Chapman, D. and Penn, A. (1999). Sketching a Virtual Environment: Modeling Using Line Drawing Interpretation. In *Proceedings of the ACM symposium on Virtual reality software and technology, London, United Kingdom* (pp. 155-161). New York: ACM Press.

Tuzmen, A. (2000). Collaborative Building Design. In Dirk Donath (Ed.), *Proceeding of the 18th Conference on Education in Computer Aided*

Architectural Design in Europe, 2000 (pp. 93-98). Bauhaus Universität, Weimar, Germany.

Ucelli, G., Conti, G., Lindsay, M. and Ryder, G. (2000). From "Soft" to "Hard" Prototyping: A Unique Combination of VR and RP for Design. In Robin Holland (Ed.), *Proceedings of the seventh UK VR-SIG (UK Virtual Reality Special Interest Group) 2000 conference in Glasgow, Sept. 2000* (pp. 1-9). UK: UK VR-SIG.

University of Strathclyde, (2002). The Virtual Environmental Laboratory. Retrieved 22nd September 2002 from the World Wide Web: http://www.strath.ac.uk/Departments/Architecture/VEL/index.html

Valin, S., Francu, A., Trefftz, H. and Marsic, I. (2001). Sharing Viewpoints in Collaborative Virtual Environments. In *34th Annual Hawaii International Conference on System Sciences (HICSS-34)-Volume 1, January 2001, Maui, Hawaii* (p. 1032). Retrieved 28th August 2002 from the World Wide Web: http://citeseer.nj.nec.com/valin00sharing.html

Vernor Vinge (1981), *True Names, Binary Star #5*. New York: Dell Publishing Company.

VR-Glasgow (2002). Retrieved 28th August 2002 from the World Wide Web: http://iris.abacus.strath.ac.uk/gary/Miscellaneous/kiosk/

Walser, R. (1992). Autodesk Cyberspace Project. *Mondo 2000, 1992*. New York: HarperCollins.

Web3D Consortium (2002). Xj3D Open Source VRML/X3D Toolkit. Retrieved 30th September 2002 from the Word Wide Web: http://www.web3d.org/TaskGroups/source/xj3d.html

Wedeles, L. (1966). Prof. Nelson Talk Analyzes P.R.I.D.E. *Vassar Miscellany News February 3, 1965*. Retrieved 13th June 2002 from the World Wide Web: http://iberia.vassar.edu/~mijoyce/MiscNews_Feb65.html

West, A. and Hubbold, R. (2001). System Challenges for Collaborative Virtual Environments. In Elizabeth F. Churchill, David N. Snowdon and Alan J.

Munro (Eds.), *Collaborative Virtual Environments: Digital Places and Spaces for Interaction, 2001* (pp. 43-54). London: Springer.

Whitaker, R. (1996). Collaborative Interior Design. Retrieved 7th August 2002 from the World Wide Web: http://www.cs.iupui.edu/~tuceryan/research/AR/interior_design.html

Williams, A. and Cowdroy, R. (2002). How Designers Communicate Ideas to Each Other in Design Meetings. In D. Marjanovic (Ed.), *Proceedings of the Design 2002, 7th International Design Conference 2002* (pp. 947-952). Zagreb, Croatia: Faculty of Mechanical Engineering and Naval Architecture, Zagreb and The Design Society, Glasgow, UK.

Wittmann, A. (2000). The 10 Most Important Products of the Decade. *Network Computing Oct. 2000*. Retrieved 28th June 2002 from the World Wide Web: http://www.networkcomputing.com/1119/1119f1products_intro.html

Yeung, C. (1997). A Web-Based VRML Collaborative Design Tool for Architecture Studies. In B. Martens, H. Linzer and A. Voigt (Eds.), *Proceedings of 15th eCAADe conference, 1997* (pp. 92-100). Vienna, University of Technology. Retrieved 07th August 2002 from the World Wide Web: http://info.tuwien.ac.at/ecaade/proc/yeung1/paper/ecaade.htm

Appendix A: The Content of the Chat during the Experiment

A.1 Overview of the Content of the Chat

The following pages report the text of the chat recorded during the experiment described in Chapter 7. The first part of the text was recorded during the introductory session (lines 1 - 20), the second during the experiment (lines 21 – 171).

Four users are registered in the session, three students named Ross, Eddie and Christoph, and one of the authors, Giuseppe, was included to supervise the outcome of the experiment. He intervenes at points to announce the beginning of the experiment (line 21), to communicate the crash of the application run on the Sgi machine (lines 37, 39, 154), to warn students of the imminent end of the experiment (lines 149, 151, 153 and 157, 158) and to announce the end of the test (lines 167, 168).

From the analysis of the text it is possible to appreciate some interesting features of the system, for example line 39 stresses the network persistence mechanism described in Section 6.2.4.

The text also contains suggestions for improvements highlighted by the students. Specifically need to use a virtual compass or a grid to help localise the object is reported at lines 72 and 83. The development of a mechanism to place objects perpendicularly is suggested at line 73. The further development of the navigation mode is suggested at line 83.

Finally the text shows the relative unreliability of JCAD-VR on the Sgi Irix platform. While the two PCs used in the experiment proved to be reliable throughout the Sgi Onyx2 crashed three times (lines 37, 150, 154).

A.2 The Text of the Chat

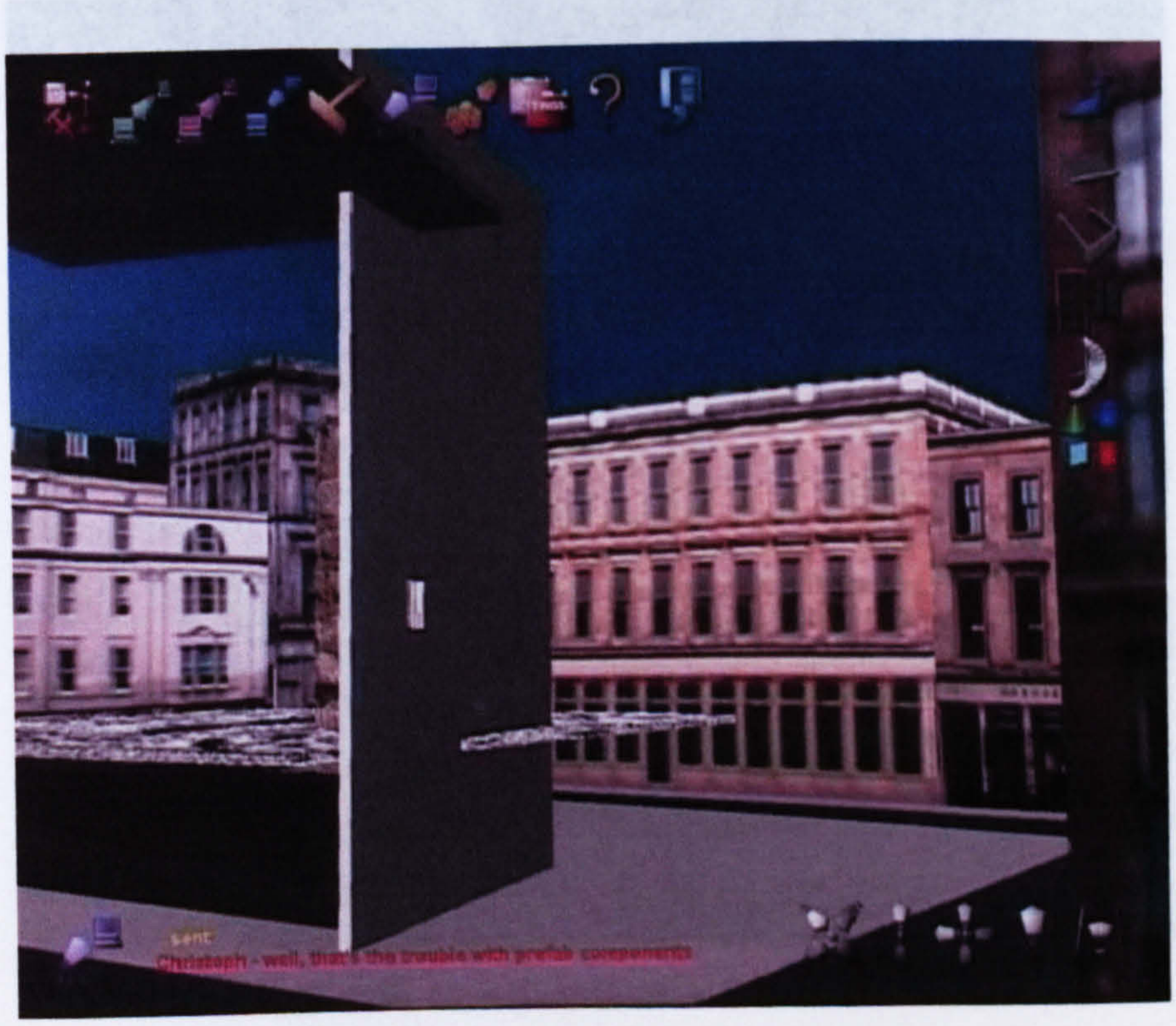
1. Nothing received
2. **Eddie** - hi we are here...giuseppe ??????????????& ed...
3. **Ross** - hello
4. **Christoph** - open chat window for sketching, please
5. **Eddie** - are you asleep Christoph ?
6. **Christoph** - i am not sleeping, just busy

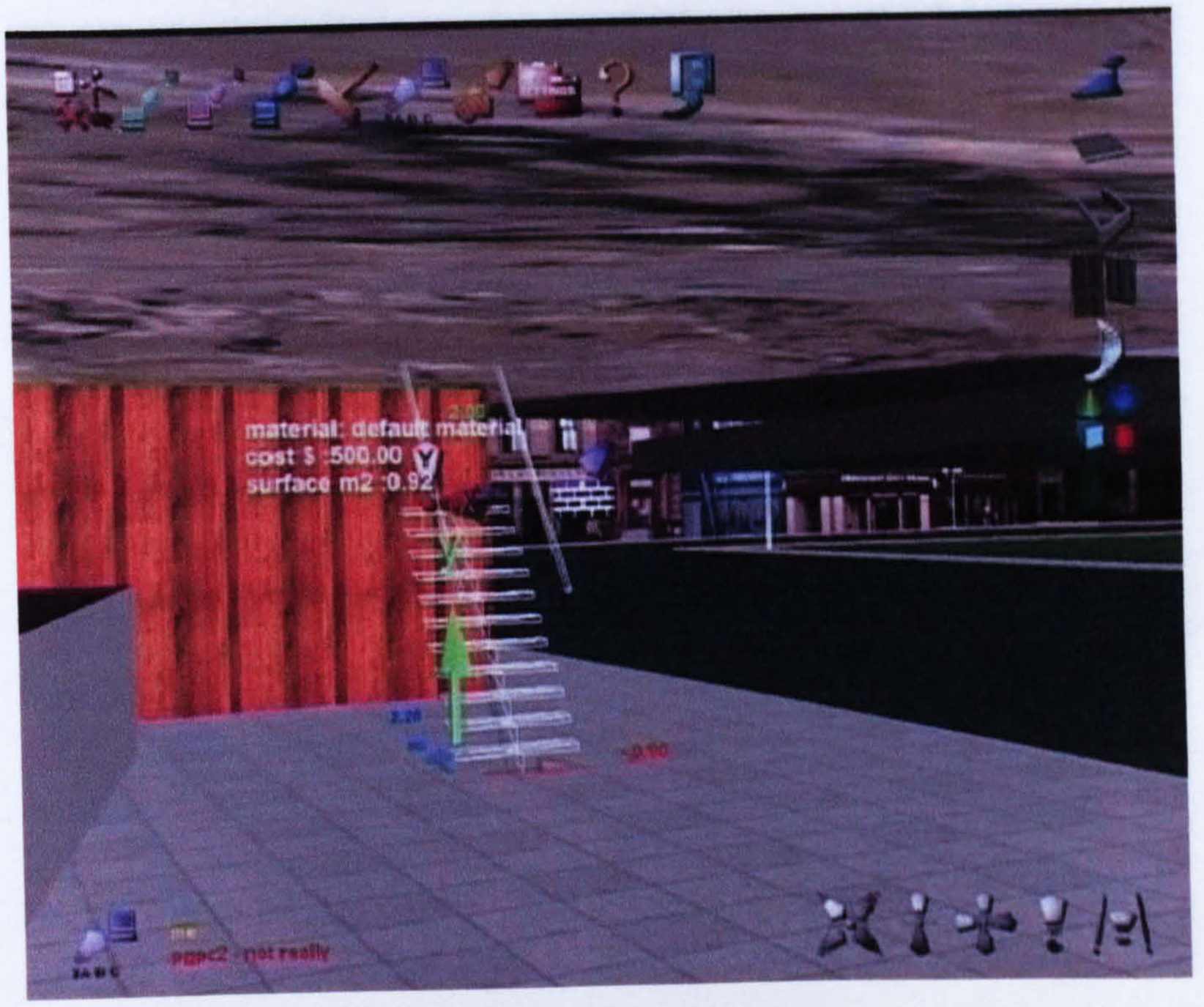
7. **Christoph** - who is sketching?\\
8. **Eddie** - not me
9. **Christoph** - was me, Christoph
10. **Christoph** - was this somebody's design?????
11. **Christoph** - eddie are you Edward ??
12. **Christoph** - eddie is that you?? sketching with red??
13. **Christoph** - i am white, Christoph
14. **Christoph** - who created stair??\anybody answere me\
15. **Ross** - it wasnt me
16. **Christoph** - cleaned up the mess ?:)
17. **Ross** - who is cleaning up the mess??
18. **Eddie** - you didnt like it??
19. **Eddie** - um probably
20. **Christoph** - i am talking about the geometry mistake??
21. **Giuseppe** - hi guys , shall we start?? go...
22. **Ross** - ?ok
23. **Edward** - ok
24. **Christoph** - ok
25. **Ross** - ok
26. **Eddie** - ?how about a big wall??
27. **Ross** - thats ok with me
28. **Ross** - ok
29. **Christoph** - hi, i',m there ?:)
30. **Ross** - i put a cylinder on the coner
31. **Eddie** - what is it made of??
32. **Eddie** - this is a kiosk..... how big??
33. **Eddie** - is it selling hot dogs??
34. **Ross** - □is that too big??????????????????
35. **Eddie** - anybody??
36. **Eddie** - probably
37. **Giuseppe** - guys Christoph crashed he's re-booting
38. **Ross** - ok so what will do do then??
39. **Giuseppe** - guys if someone crashes keep on working it's no problem. Once on-line again the system loads the content of the world automatically
40. **Christoph** - hi, guys, guess who's back??!
41. **Ross** - its looking very abstract
42. **Christoph** - i would put a roof on it, like this, check sketchpad
43. **Ross** - ??????????????????????where will the entrance be??
44. **Christoph** - na, the whole thing is the kiosk, like a pavillion
45. **Christoph** - have to check the roof it does look a little bit nasty
46. **Eddie** - what's going on??
47. **Ross** - ?are theyre any windows??
48. **Christoph** - well, that's the trouble with prefab components
49. **Christoph** - where's glas?
50. **Christoph** - ross, are you ross?\
51. **Ross** - yes i tink so
52. **Ross** - opps i del the wall
53. **Christoph** - sometimes is tricky, where are u?
54. **Eddie** - is this still a small kiosk??
55. **Ross** - not any more ?:?)
56. **Christoph** - na, not really, small elements are tricky, though
57. **Christoph** - can u copy elements?????
58. **Christoph** - barcelona pavillion i guess ;)
59. **Ross** - do we want steps up to the platform??????????????
60. **Eddie** - yes why not.. big ones
61. **Christoph** - why not, give it a shot

62. **Eddie** - steps??
63. **Ross** - ?stpes or st\irs??????????
64. **Eddie** - ones like this
65. **Christoph** - can we reduce the height of the cylinder, please
66. **Ross** - ok
67. **Eddie** - what should the stairs be made from??
68. **Christoph** - you can't change materials of symbols
69. **Eddie** - no
70. **Christoph** - na, can't apply material to stairs
71. **Eddie** - i think we need to do some site analysis
72. **Christoph** - where;s south, would really like a grid or something
73. **Christoph** - hard to get objects perpenticular
74. **Christoph** - what's that ?
75. **Ross** - what??
76. **Christoph** - that grey misobject,
77. **Christoph** - the design is a mess, gonna be a labyrint
78. **Eddie** - where is the entrance??
79. **Christoph** - guess?i don't know it's on the lifeft side, from my point of view, i will highlight it
80. **Christoph** - that??!!!!
81. **Christoph** - you see?
82. **Eddie** - that roof is floating
83. **Christoph** - navigation mode would be really nice, like in airplanes, like 2 o'clock high, and a definite no9rthpoint
84. **Ross** - do we need the large wall on the left??
85. **Christoph** - where?
86. **Ross** - there
87. **Eddie** - minimalists
88. **Christoph** - who's creatring these massive walls
89. **Christoph** - all the time?
90. **Eddie** - i made one of them.
91. **Ross** - i made 1
92. **Christoph** - skyhigh, lets go 3d
93. **Christoph** - anybody, please answere, what's that on the left?
94. **Christoph** - must be a wall with negatyive thickness including windows?
95. **Ross** - the wall that i just deleted??
96. **Ross** - ?the roof is floating now
97. **Eddie** - i think it should be lower
98. **Ross** - should it sit on the cylinder??
99. **Eddie** - is the roof a creche then??
100. **Christoph** - youyoulook at it, kiosk??
101. **Eddie** - there is no entrance to the park??!?
102. **Ross** - did we decide on an entrance yet??
103. **Eddie** - no , but it needs one i think
104. **Ross** - where??
105. **Eddie** -
106. **Eddie** - nice door sould be wider though
107. **Ross** - □???????is that ok??
108. **Eddie** - like it ys
109. **Christoph** - other side towards the buildings
110. **Ross** - what do we have happening on the street??
111. **Ross** - ???
112. **Eddie** - what have i missed??
113. **Ross** - what is happening??
114. **Eddie** - i thi9nk christoph is going mad
115. **Ross** - is it going to be open to the street??
116. **Eddie** - is it ??

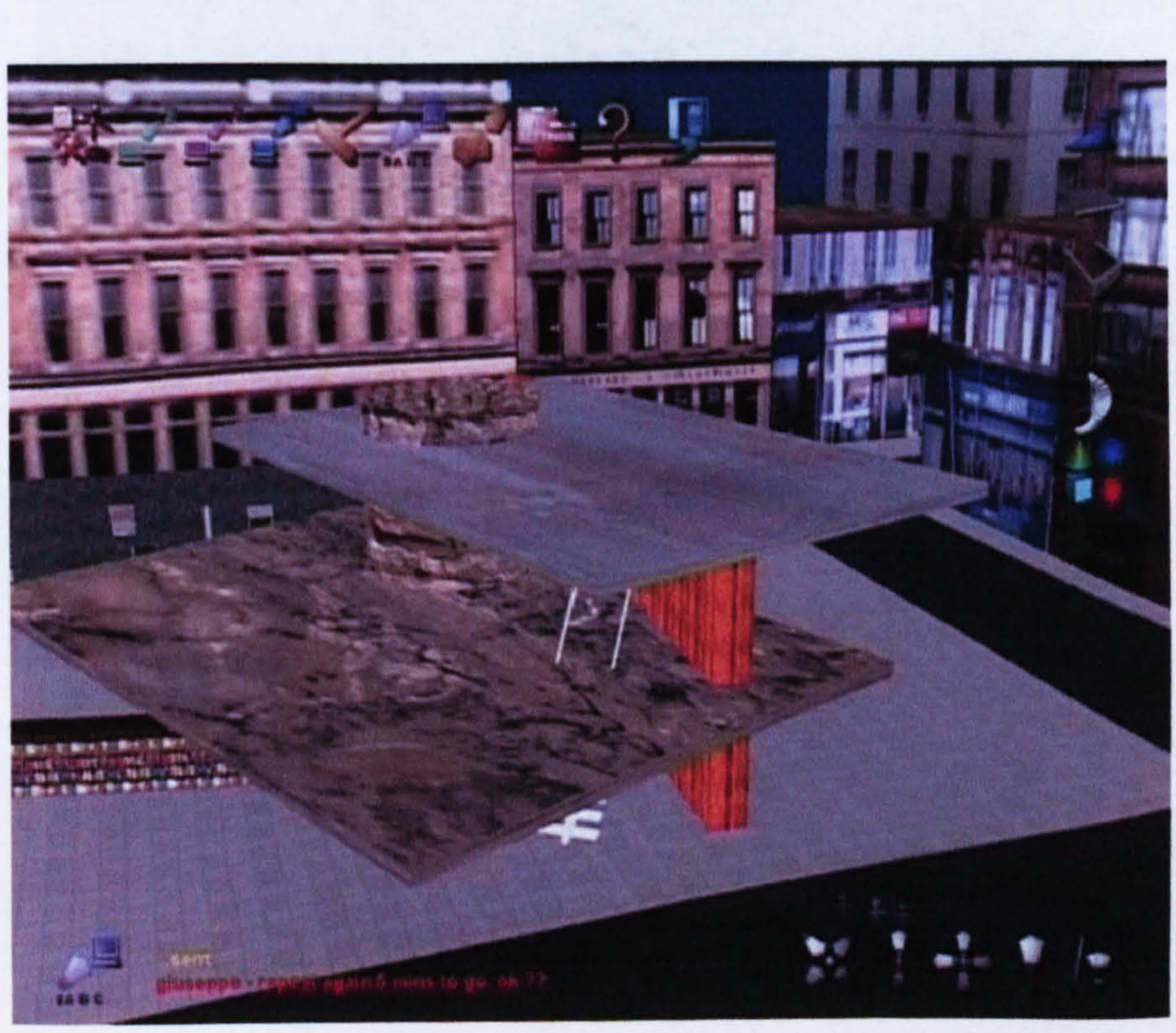
117. **Ross** - ?what do u think??
118. **Eddie** - what happened to the wall with the windows??
119. **Eddie** - timber?!
120. **Ross** - for the wall on left??
121. **Eddie** - yeah
122. **Ross** - pannels???
123. **Eddie** - yes thats good
124. **Eddie** - did anyone really ant the slide??
125. **Ross** - not me
126. **Ross** - should the back wall be higher??
127. **Eddie** - just a bit
128. **Eddie** - maybe make a balustrade to the roof
129. **Eddie** - or maybe not
130. **Christoph** - single storey for a kiosk,please
131. **Eddie** - lighten up Christoph
132. **Eddie** - it's falling water now
133. **Ross** - ?:?)
134. **Eddie** - how about it doubles as a beer garden??
135. **Ross** - in a kiosk??
136. **Eddie** - why not??
137. **Ross** - should there be something on the leftside??
138. **Eddie** - hmmm just a small intervention
139. **Eddie** - whoose is the cylinder??
140. **Ross** - which 1????
141. **Eddie** - the big un
142. **Ross** - me
143. **Ross** - should it go??
144. **Eddie** - i think we should buy up some of the park for a new site then
145. **Eddie** - put it on stilts
146. **Eddie** - where is Christoph??
147. **Eddie** - i was just getting used to that
148. **Eddie** - not really
149. **Giuseppe** - guys 10 min to go from now
150. **Christoph**: ?i crashed again
151. **Giuseppe** - got it ???
152. **Eddie** - are we keeping this monstrosity??
153. **Giuseppe** - I repeat 10 mins to go from now ok ?
154. **Giuseppe** - chris is crashed again...sgi is bit unstable
155. **Ross** - guys hang on 4 a min
156. **Christoph** - hi guys, finally back
157. **Giuseppe** - ok now everything is fine u have 5 min
158. **Giuseppe** - repeat again 5 mins to go, ok ??
159. **Eddie** - ok
160. **Christoph** - ok
161. **Ross** - ?what is in the middle of the park now???
162. **Christoph** - back again
163. **Ross** - ?ok
164. **Christoph** - are we not takiung that alittle bit far?
165. **Eddie** - is that the motorway??
166. **Ross** - thats what i thought
167. **Giuseppe** - I'm deleting all objects now...
168. **Giuseppe** - that's you guys...thankx very much 4 your time !!
169. **Ross** - ok
170. **Christoph** - 0k
171. **Eddie** - ok!

Appendix B: Screenshots from the Experiment

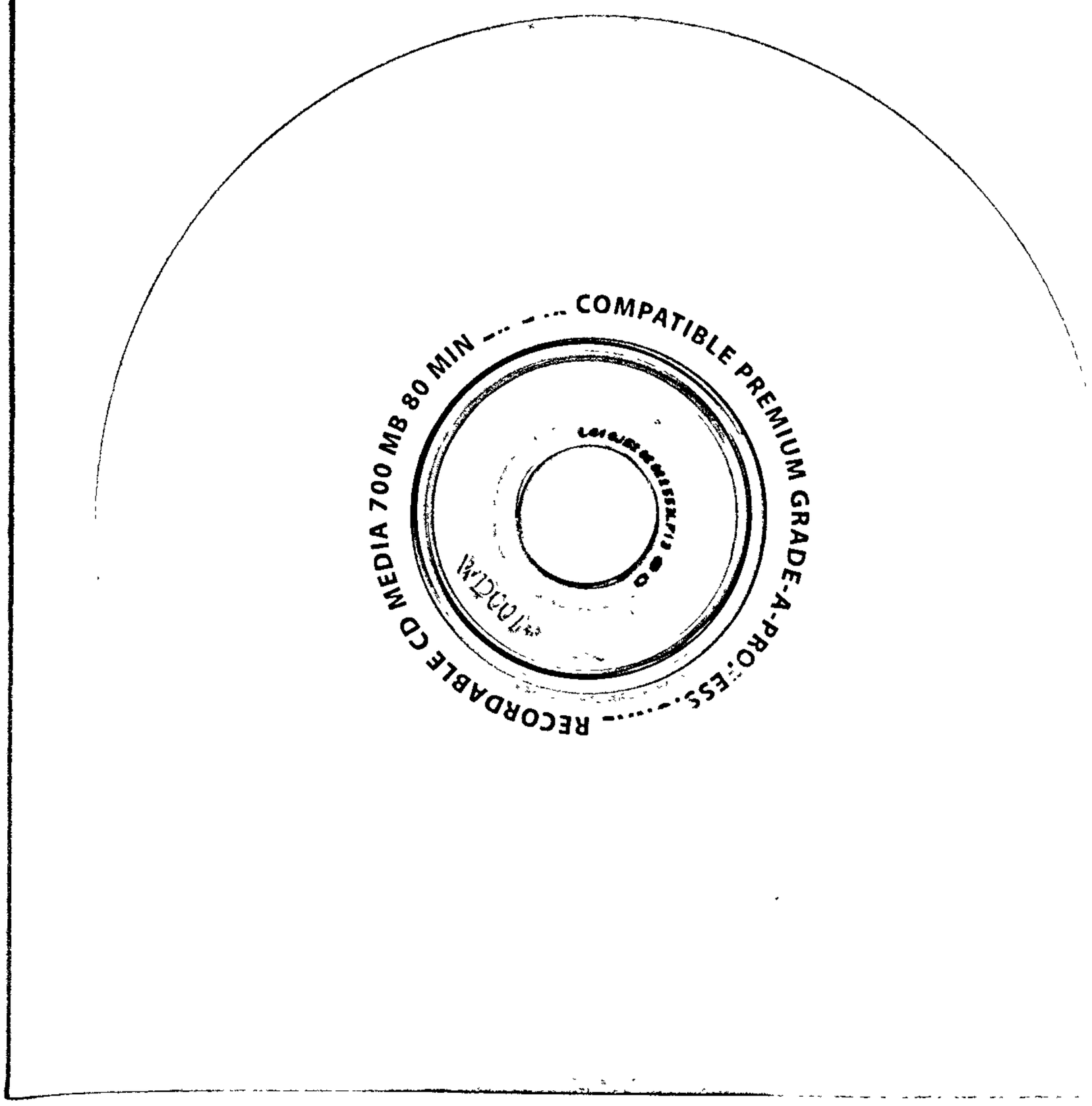



material: default material
cost $:500.00
surface m2 :0.92
pppc2 : not really


me:



Appendix C: The CD-ROM with the Video of the Experiment



Appendix D: Specification of Hardware and Software used for the Experiment

D.1 Specification Computer n.1 (client)

Hardware

- Model: Dell Precision 530
- Processor(s): 2 x Intel® Xeon™ 1.4 GHz
- Memory: 512 MB Ram
- Graphic Card: nVidia Quadro2 EX with 32Mb Video Memory

Software

- Operative System: Windows 2000 SP2
- Java™ 2 SDK Environment, Standard Edition 1.3.1_01
- Java 3D™ 1.2.1_03 SDK (OpenGL Version)

D.2 Specification Computer n.2 (client)

Hardware

- Model: Viglen Genie
- Processor(s): Intel® Pentium™ III 933 MHz
- Memory: 256 MB Ram
- Graphic Card: nVidia RIVA TNT2-Model64 with 32Mb Video Memory

Software

- Operative System: Windows 2000 SP2
- Java™ 2 SDK Environment, Standard Edition 1.3.1_01
- Java 3D™ 1.2.1_03 SDK (OpenGL Version)

D.3 Specification Computer n.3 (client)

Hardware

- Model: Sgi Onyx2
- Processor(s): 12 x MIPS R12000 400MHz
- Memory: 6GB
- Graphic Card: 2 x InfiniteReality

Software

- Operative System:
- Java™ 2 SDK 1.3.1 for Sgi Irix: MR Release
- Java 3D™ 1.2.1 SDK for Sgi Irix: MR Release

D.4 Specification Computer n.4 (server)

Hardware

- Model: ® O2
- Processor(s): MIPS R5000 300MHz
- Memory: 128 MB
- Graphic Card: CRM graphics

Software

- Operative System:
- Java™ 2 SDK 1.3.1 for Sgi Irix: MR Release
- Java 3D™ 1.2.1 SDK for Sgi Irix: MR Release

D.5 External Libraries used in JCAD-VR

- Prominence library (Hughes et al., 1997)
- Xj3D loader release 3 (Web3D Consortium, 2002)

Appendix E: Questionnaires Proposed for the Cross-University Experiment

QUESTIONNAIRE - Group 1 - (*JCAD-VR Collaborative)*

Part 1:

- Session: ☐ Morning
  ☐ Afternoon
- Year of study:.....
- What kind of software do you normally use? ☐ Microsoft Office

  ☐ CAD/CAAD packages
  If selected please specify:

.....

.....

☐ Other
If selected please specify:

.....

.....

- Have you had any experience with *Virtual Reality* before?
  - ☐ Yes, many times
  - ☐ Yes, few times
  - ☐ Yes, once
  - ☐ No

Part 2:

1. How do you find navigating within the virtual environment using this software?
   - ☐ Very easy
   - ☐ Easy
   - ☐ Fairly easy
   - ☐ Difficult – **specify why**.....

.....

.....

☐ Very difficult – specify why.....

.....

.....

2. How did you find creating objects using this software?

☐ Very easy

☐ Easy

☐ Fairly easy

☐ Difficult – specify why.....

.....

.....

☐ Very difficult – specify why.....

.....

.....

3. How did you find modifying objects using this software?

☐ Very easy

☐ Easy

☐ Fairly easy

☐ Difficult – specify why.....

.....

.....

☐ Very difficult – specify why.....

.....

.....

4. How did you find working with your partner simultaneously?

☐ Very easy

☐ Easy

☐ Fairly easy

☐ Difficult – specify why.....

.....

.....

☐ Very difficult – specify why.....

.....

.....

5. Did you find it interesting to work with a partner over the net?

   ☐ Yes, a lot

   ☐ Yes

   ☐ Not so much – specify why.....

.....

.....

.....

   ☐ No – specify why.....

.....

.....

   ☐ No, absolutely – specify why.....

.....

.....

6. In your view did you and your partner have enough exchange of information in order to accomplish the design task?

   ☐ Yes, a lot

   ☐ Yes

   ☐ Not so much – specify why.....

.....

.....

   ☐ No – specify why.....

.....

.....

   ☐ No, absolutely – specify why.....

.....

.....

7. Do you think that your partner could have misinterpreted your design ideas during the collaboration?

   ☐ Yes, many times we had problems and we could not explain our ideas

Thanks for your kind help. The information that you have provided will be very useful for us.

QUESTIONNAIRE - Group 2 - (*JCAD-VR Stand-alone*)

Part 1:

- Session: ☐ Morning
  ☐ Afternoon
- Year of study:.....
- What kind of software do you normally use? ☐ Microsoft Office
  ☐ CAD/CAAD packages
  If selected please specify:

.....

.....

☐ Other
If selected please specify:

.....

.....

- How frequently do you use e-mails:
  - ☐ Very often
  - ☐ Often
  - ☐ Sometimes
  - ☐ Seldom
  - ☐ Never

- Have you had any experience with *Virtual Reality* before?
  - ☐ Yes, many times
  - ☐ Yes, few times
  - ☐ Yes, once
  - ☐ No

Part 2:

1. How do you find navigating within the virtual environment using this software?

   ☐ Very easy
   ☐ Easy
   ☐ Fairly easy
   ☐ Difficult – specify why.....
   .....
   .....
   ☐ Very difficult – specify why.....
   .....
   .....

2. How did you find creating objects using this software?

   ☐ Very easy
   ☐ Easy
   ☐ Fairly easy
   ☐ Difficult – specify why.....
   .....
   .....
   ☐ Very difficult – specify why.....
   .....
   .....

3. How did you find modifying objects using this software?

   ☐ Very easy
   ☐ Easy
   ☐ Fairly easy
   ☐ Difficult – specify why.....
   .....
   .....
   ☐ Very difficult – specify why.....

.....

.....

4. How did you find the collaboration with your partner?

   ☐ Very easy

   ☐ Easy

   ☐ Fairly easy

   ☐ Difficult – specify why.....

.....

.....

   ☐ Very difficult – specify why.....

.....

.....

5. Did you find it interesting to work with a partner over the net?

   ☐ Yes, a lot

   ☐ Yes

   ☐ Not so much – specify why.....

.....

.....

   ☐ No – specify why.....

.....

.....

   ☐ No, absolutely – specify why.....

.....

.....

6. In your view did you and your partner have enough exchange of information in order to accomplish the design task?

   ☐ Yes, a lot

   ☐ Yes

   ☐ Not so much – specify why.....

.....

.....

☐ No – specify why.....

.....

.....

☐ No, absolutely – specify why.....

.....

.....

7. In your view is the collaborative method you have used in this experiment (software+e-mails) a feasible approach for collaborative design?

   ☐ Yes, I found it effective

   ☐ Yes, but.....

.....

.....

   ☐ No, but.....

.....

.....

   ☐ No, I found it unpractical

8. Do you think that your partner could have misinterpreted your design ideas during the collaboration?

   ☐ Yes, many times we had problems and we could not explain our ideas

   ☐ Yes, but eventually we succeeded in explaining our ideas

   ☐ No, but the communication media available were just enough to show our ideas

   ☐ No, the communication media available allowed us to show our ideas

9. Do you think that this software could help in proposing design solutions at an initial, conceptual, stage in relation with the urban context?

   ☐ Yes, a lot

   ☐ Yes

   ☐ Not so much – specify why.....

.....

.....

☐ No – specify why..

....

....

☐ No, absolutely – specify why...

....

....

10. We warmly encourage any kind of comment on the software and on the experiment itself:

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

Thanks for your kind help. The information that you have provided will be very useful for us.

QUESTIONNAIRE - Group 3 - (*Sketching*)

Part 1:

- Session: ☐ Morning
  ☐ Afternoon
- Year of study:.....

- Are you confident in using:

a printer: ☐ Yes
☐ Fairly
☐ No

a scanner: ☐ Yes
☐ Fairly
☐ No

- How frequently do you use e-mails:
  ☐ Very often
  ☐ Often
  ☐ Sometimes
  ☐ Seldom
  ☐ Never

Part 2:

1. How did you find expressing your design ideas and solutions sketching?
   ☐ Very easy
   ☐ Easy
   ☐ Fairly easy
   ☐ Difficult – specify why.....
.....
.....
   ☐ Very difficult – specify why.....

.....

.....

2. How did you find the collaboration with your partner?

   ☐ Very easy

   ☐ Easy

   ☐ Fairly easy

   ☐ Difficult – specify why.....

.....

.....

   ☐ Very difficult – specify why.....

.....

.....

3. Did you find it interesting to work with a partner over the net?

   ☐ Yes, a lot

   ☐ Yes

   ☐ Not so much – specify why.....

.....

.....

   ☐ No – specify why.....

.....

.....

   ☐ No, absolutely – specify why.....

.....

.....

4. In your view did you and your partner have enough exchange of information in order to accomplish the design task?

   ☐ Yes, a lot

   ☐ Yes

   ☐ Not so much – specify why.....

.....

.....

☐ No – specify why.....

.....

.....

☐ No, absolutely – specify why.....

.....

.....

5. Do you think that your partner could have misinterpreted your design ideas during the collaboration?
   - ☐ Yes, many times we had problems and we could not explain our ideas
   - ☐ Yes, but eventually we succeeded in explaining our ideas
   - ☐ No, but sketches were just enough to show our ideas
   - ☐ No, sketches allowed us to show our ideas

6. In your view is the collaborative method you have used in this experiment (sketches+e-mails) a feasible approach for collaborative design?
   - ☐ Yes, I found it effective
   - ☐ Yes, but.....

.....

.....

.....

☐ No, but.....

.....

.....

☐ No, I found it unpractical

7. We warmly encourage any kind of comment on the collaborative method and on the experiment itself:

.....

.....

.....

.....

.....

.....

Thanks for your kind help. The information that you have provided will be very useful for us.

Appendix F: Notes for Tutors and Troubleshooting

F.1 Notes for tutors

A General Record

In both universities it would be appropriate to record the experiment with a video camera (preferably a digital one) and to take some pictures with a digital camera.

Other Notes

- It would be useful to set up a communication line between the tutors via ICQ or similar chat-lines during the experiment (addresses would be required).
- Both universities have to provide email access for groups 2 and 3, and also a printer, a scanner at A4, A4 paper and sketching materials for group 3.
- At the end of **each session** tutors should copy the content of the following folders where all the screenshots, back-up files, final images and emails are located:

  1. *JCAD-VR\automatic_save*,
  2. *JCAD-VR\scene*
  3. *JCAD-VR\capture*
  4. Folder with the emails of group 2
  5. Folder with the emails of group 3

Before deleting the files contained in these directories their content has to be saved in another directory in order to start the second session with empty folders. This will avoid mixing up the files produced by the different students in the two sessions.

F.2 Notes and Troubleshooting for group 1

Loading jcad Files when Starting JCAD-VR

JCAD-VR will automatically load the environment to be shared by the students working synchronously. This means that new objects created within JCAD-VR will be visualized in every computer joining the concurrent design session. These are the instructions to follow to load the scene at the beginning of the exercise:

1. Start JCAD-VR
2. During the loading of JCAD-VR a window will appear to allow the selection of a VRML file, click on CANCEL
3. After few seconds the scene will be loaded, and when all the partners joining the session are ready, every object created will be visualized and shared with the others.

IMPORTANT NOTE: Before starting to create objects in the collaborative mode all the users have to be connected with the application running and connected to the server. **Make sure that all your partners in the collaborative session are already connected properly before starting to create objects**. The best way to do thsi is simply by monitoring the text of the chat; if you are able to chat with your partners you will be also able to share objects. If geometries were already created before your partners were connected we suggest you to delete all the objects by clicking on the *delete all*  icon, unless you wish to continue by sharing the objects with your partners.

Loading Files without Restarting JCAD-VR

1. Load your new file by clicking on the *load*  icon

2. If the system is in collaborative mode, once you load the file the other user connected will receive the scene. Therefore this operation has to be done **by only one person involved in the collaborative session**. Double check with your partner by chat to see if they have the scene upgraded before starting to work.

In case of the Crashing or Interruption of JCAD-VR while a Student is Working with a Partner:

1. Close JCAD-VR by clicking on the *exit*  icon or using the CTRL+C button sequence on the MS-DOS console.
2. Restart JCAD-VR
3. During the loading of JCAD-VR a window will appear to allow selection of a file, click on CANCEL
4. Wait for some seconds and the software will automatically upgrade the scene.

Saving with JCAD-VR

In order to make back-up files, JCAD-VR will automatically save students' work every 5 minutes and the files will be placed (with a name and sequential number) in the directory **JCAD-VR\automatic_save**. Nevertheless students can also save their design by clicking on the *save* icon  and by giving these names to their files: **group1_01**, **group1_02, ...group1_20** etc... in the **JCAD-VR\scene** directory.

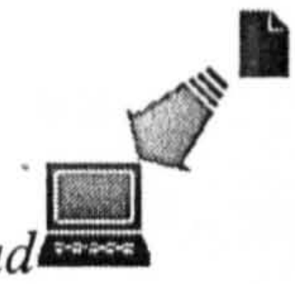
IMPORTANT LAST NOTE: *For normal reasons JCAD-VR can also crash or halt if some unexpected operations have been done. We would therefore recommend saving your work frequently.*

F.3 Notes and Troubleshooting for group 1

Loading jcad Files when Starting JCAD-VR

JCAD-VR automatically loads the environment where you have to place your design. You need only to start JCAD-VR, and the environment will appear.

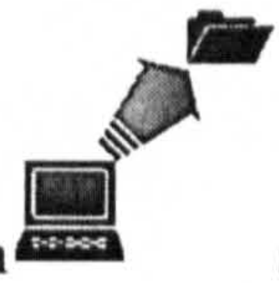
Loading Files without Restarting JCAD-VR

The loading routine of JCAD-VR is such that more than one jcad file cannot be loaded together. You can load your new file by clicking on the *load*  icon and by selecting the file to open (jcad and wrl formats are supported).

In case of the Crashing or Interruption of JCAD-VR

1. Close JCAD-VR by clicking on the *exit*  icon or using the CTRL+C button sequence on the MS-DOS console.
2. Restart JCAD-VR
3. During the loading of JCAD-VR a window will appear to allow the selection of a file, click on CANCEL
4. Load the jcad file you were working on again (for instructions on how to load a file in JCAD-VR see *Loading jcad files when starting JCAD-VR* on the previous page).

Saving with JCAD-VR

Students can save their design by clicking on the *save* icon  and by giving these names to their files: **group2_01**, **group2_02, …group2_20** etc… in the **JCAD-VR\scene** directory.

IMPORTANT LAST NOTE: *For normal reasons JCAD-VR can also crash or halt if some unexpected operations have been done. We would therefore recommend saving your work frequently.*

Appendix G: The JCAD-VR Project's Publications

The following pages include conference papers reporting on the JCAD-VR project:

1. Conti, G, Ucelli, G. and Maver, T. (2001). JCAD-VR: Java Collaborative Architectural Design Tool in VR. In H. Penttilä (Ed.), *Proceedings of Architecture Information Management, 19th Conference of eCAADe, Helsinki, 2001* (pp. 454-459). Espoo, Finland: Otamedia Oy.

2. Ucelli, G., Conti, G., Petric, J. and Maver, T. (2002). Real Experiences of Virtual Worlds. In D. Marjanović (Ed.), *Proceedings of the Design 2002, 7th International Design Conference, 2002, Cavtat* (pp. 561-566). Zagreb, Croatia: Faculty of Mechanical Engineering and Naval Architecture, Zagreb and The Design Society, Glasgow, UK.

3. Petric, J., Maver, T., Conti, G. and Ucelli, G. (2002). Virtual Reality in the Service of User Participation in Architecture. In K. Agger, P. Christiansson and R. Howard (Eds.), *Distributing Knowledge in Building. Proceedings of CIB W78 Conference, Aarhus* (pp. 217-224). Aarhus, Denmark: Aarhus School of Architecture and Centre for Integrated Design.

4. Petric, J., Ucelli, G. and Conti, G (2002). Real Teaching and Learning through Virtual Reality. In K. Koszewski and S. Wrona (Eds.), *Design e-ducation. Proceedings of the 20th Conference of eCAADe, Warsaw, 2002* (pp. 72-79). Warsaw, Poland: Drukarnia Braci Ostrowskich.

5. Conti, G., Ucelli, G. and Petric, J. (2002). JCAD-VR: a collaborative design tool for architects. In *Proceedings of the 4th international conference on Collaborative virtual environments, Bonn, Germany, 2002* (pp. 153-154). New York: ACM Press.

6. Petric, J., Ucelli, G. and Conti, G. (2002). Participatory Design in Collaborative Virtual Environments. Accepted paper at the 6th SIGRADI Conference, Caracas.

7. Petric, J. Conti, G. and Ucelli, G. (2003). Designing Within Virtual Worlds. Accepted paper at the CAAD Futures 2003 Conference, Tainan, Taiwan

Conti, G, Ucelli, G. and Maver, T. (2001). JCAD-VR: Java Collaborative Architectural Design Tool in VR. In H. Penttilä (Ed.), ***Proceedings of Architecture Information Management, 19th Conference of eCAADe, Helsinki, 2001*** **(pp. 454-459). Espoo, Finland: Otamedia Oy.**

Go to contents 16

JCAD-VR: Java Collaborative Architectural Design Tool in Virtual Reality

A Java3D based scalable framework for real-time, multi-platform VR environments

CONTI, Giuseppe; UCELLI, Giuliana; MAVER, Tom
ABACUS, University of Strathclyde, Glasgow, UK
http://www.strath.ac.uk/Departments/Architecture/

This paper proposes a framework that provides the architect with a tool that uses Virtual Reality (VR) as part of the design path. It offers the possibility to deploy a system capable of assisting the design profession during the early stages of the design process. This way VR becomes the means for a new experience where the architect can, free from constraints of the 2D world, create and manipulate the space she/he is designing.
The idea upon which JCAD-VR is being built is that all the users present in the virtual world have to be able to share the same virtual environment in a "transparent fashion" where the user interface, instead of the traditional menu/windows based layout, it is part of the virtual world itself.
The aim is to provide the designer with a tool for creating 3D-shapes in a shared VR environment, thus allowing the design to be shared as it evolves.
Keywords: *Collaborative Design, Virtual Reality, Java 3D, Distributed Environment.*

Introduction

Traditionally many architects have experienced the need to prove their design proposals using physical models: even the most accurate 2D paper representations are usually not suitable enough to explain and transmit the complexity of some architectonic ideas.

If the use of a third dimension is nowadays part of the daily practice, the "CAAD community" is only now experiencing the move from static representation, based on 2D renderings or pre-recorded animations (considered as a sequence of 2D images), to dynamically generated 3D representations. Real-time navigation and interaction, typical of VR environments, provide just that fluent interface and that entirety in the exploration of the design proposal that is the main lack in all CAD packages commonly in use.

Furthermore, the increasing growth of computational resources and hardware power eases the access to desktop VR applications making it a truly feasible approach in everyday practice.

Although VR is a quite mature technology, it is seldom utilised throughout the design process: often in fact it is just used as a more powerful presentation technique. Moreover, the recent growth of network-based virtual communities and the use of avatars have brought a new level of complexity to the meaning of virtuality, providing the technology for remote presence and collaborative experiences.

Background

In two case studies part of a previous research project of the authors (Ucelli, Conti, Lindsay and Ryder, 2000), the research team worked closely with both architects and engineers experimenting the use of VR in a real ongoing project.

During the creation of 3D models continuous access to all the information regarding the design progress and changing priorities were provided. Architects and engineers were invited to evaluate and discuss the project in a Virtual Reality lab.

The outcome of the research confirmed what other authors had stated (Dorta and LaLande, 1998): "the design process is made up of two activities which are graphic ideation or conceptual design and communication [...] the first is a formative process dealing with creating and evolving ideas [active phase]; the second is a descriptive process aiming at presenting to others fully-formed ideas [passive phase] "[1].

Unfortunately the result of the experience, if on the one side it has proved the huge potential of VR for reducing designs "flaws", on the other side it has highlighted that the present use of VR is limited only to the last phase of the design process, what we call the passive phase, when the visualization takes place and the result of the design process is eventually shown to the client (fig 1).

It is evident from the Figure 1 that the use of modeling and Rapid Prototyping (RP) is confined, with only a few exceptions (such in the innovative architectural practice of Frank O. Gehry) at the end of the creation process after which all the design choices have been taken.

The idea upon which the JCAD-VR framework is founded is to anticipate the use of VR within the active phase thus taking full advantage of the technology. The aim is to provide the designer with a tool for creating 3D-shapes in a shared VR environment, thus allowing the design to be shared as it evolves. This paper will report the present state of the JCAD-VR framework and will highlight its future development.

JCAD-VR: a framework

The idea upon which JCAD-VR is being built is that all the users present in the virtual world have to be able to share the same virtual environment in a "transparent fashion" where the user interface (UI), instead of the traditional menu/windows based layout, it is part of the virtual world itself.

The entire project is based on client-server architecture where every user logs into a virtual world and starts sharing designing tasks with other users. The entire structure is organised in an object-oriented

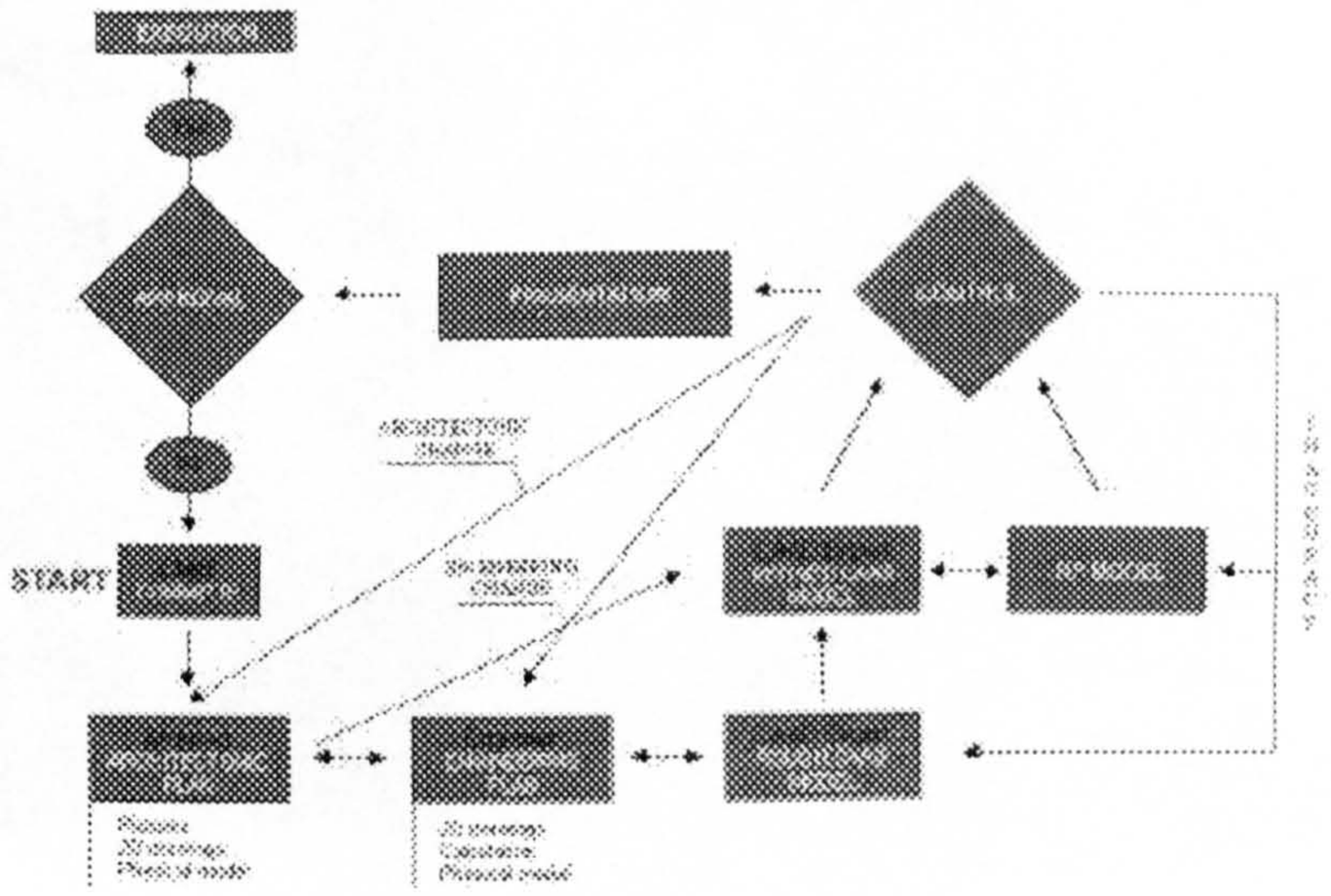


Figure 1. The role of visualisation inside the design process where the graphic ideation (active phase) is represented with the architectonic and engineering planning and the communication (passive phase) is represented by the CAAD/RP section.

fashion, where each module is able to fulfill to a certain task and it is independently coded. This approach has allowed the delivery of an initial functioning core of the JCAD-VR system whose capabilities will be expanded in the near future with several modules currently under development (fig 2).

The entire framework is handling the virtual environment through two closely connected parts: a 3D engine and a services unit each made of different modules. In a human body analogy the former might represent the heart while the latter might be considered as a nervous system.

The 3D engine unit

The 3D engine is the broad part of the framework that handles all the information regarding the "visible" aspects of the virtual world. It includes the code necessary to create and modify geometric entities (**geometry core**), to show the interface (**interface core**) and to deal with several different display devices (**visual core**).

The first module of the **geometry core** handles the creation of 2D and 3D objects: the last ones being both geometric primitives (cones, boxes, spheres etc.) and architectural entities (walls, slabs etc.). Although from the visualisation point of view a wall might be seen as a box, the system treats it in a completely different way. In fact while a box is just regarded as a simple shape without any further quality, the wall instead is handled as an entity owning "topological" properties: it is first of all made of two different surfaces (internal and external faces) and a core. It can be the parent of another object (such as a window or a door) and it can hold other types of information such as number of windows attached to it.

Quite obviously the *geometry module* will also provide the means for attaching materials to objects and add lights and objects from a library to the virtual world through the *database module*. Further development will implement real-time shape recognition routines. This would allow the user the freedom of drawing shapes in the virtual world that

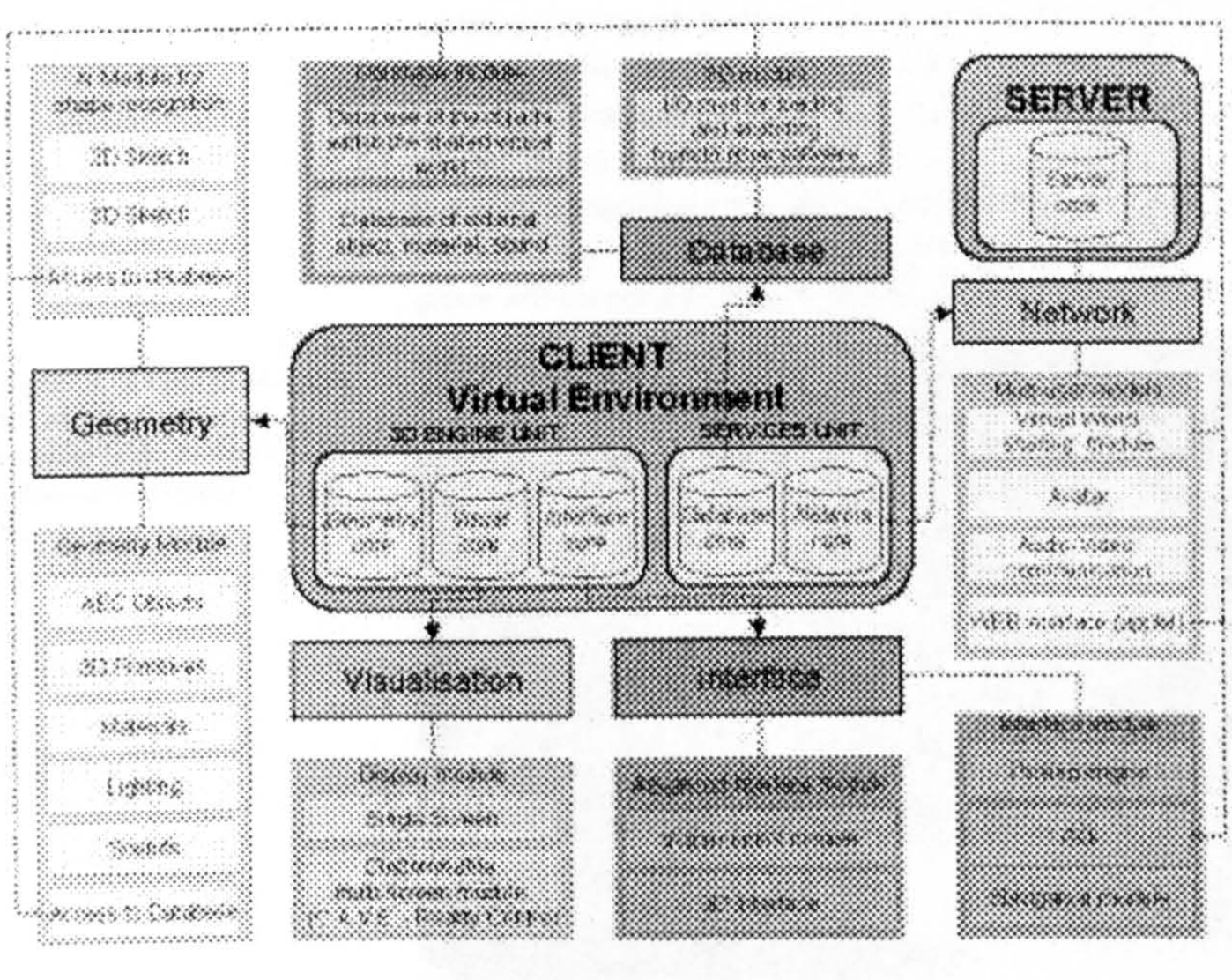


Figure 2. The general overview of the different modules

would be converted in 3D objects by the system. The complexity of such a process requires some Artificial Intelligence (AI) routines and it has convinced the authors to code the AI module only in a second stage of development.

The **interface core**, as the name itself says , obviously takes care of the graphic user interface (GUI). As previously mentioned the system aims to achieve an interface as "transparent" as possible. Here the term "transparency" is to be considered as the interface which is not a separate part of the 3D world but as an integral part of the virtual world itself. The idea behind it is that instead of the traditional menus and toolbars the user is immersed in an environment providing the means for the interaction itself. 3D menus pop up showing 3D icons and the 3D menus themselves can be moved for the convenience of the user. Visual feedback is provided, for example, in the form of rulers showing the size of objects or 3D icons showing the operation to be done on the object. A voice driven interface is due to be coded to help push further the level of "transparency" and enhance user friendliness of the interface. For the sake of completeness a traditional window based control panel is provided for advanced settings. On the first functioning core traditional pointing devices are used but support for 6-degree of freedom/virtual glove will be coded in the next future.

The **visual core** is the part of the framework that allows interfacing with the visualization devices. For the sake of flexibility the entire framework has been coded in a multi-platform language (Java). The obvious computational constrains imposed by the use of different hardware is solved by creating a structure that is flexibly scalable and can deliver images for a range of viewing devices, from the simple desktop monitor to the more complex tessellated screen [2] for immersive environments. The user can switch between two different modes according to the machine it is running on. When the software is loaded the user is asked to choose whether to work on a single screen or in multiple screen mode. At the present stage, on common PCs the video card displays the virtual world on a traditional window or on full screen. The system also runs on a Sgi supercomputer whose display is a Reality Centre. The internal architecture of this module is entirely flexible such that it might be easily adapted to allow use of other VR devices (C.A.V.E, H.M.D. etc.) (fig 3).

The services unit

The services unit is the part of the framework that handles all the information regarding the "management" of the virtual world. It is the backbone of the interconnection between users: it manages network connections and exchange of data between users (**network core**), it keeps tracks of the state of

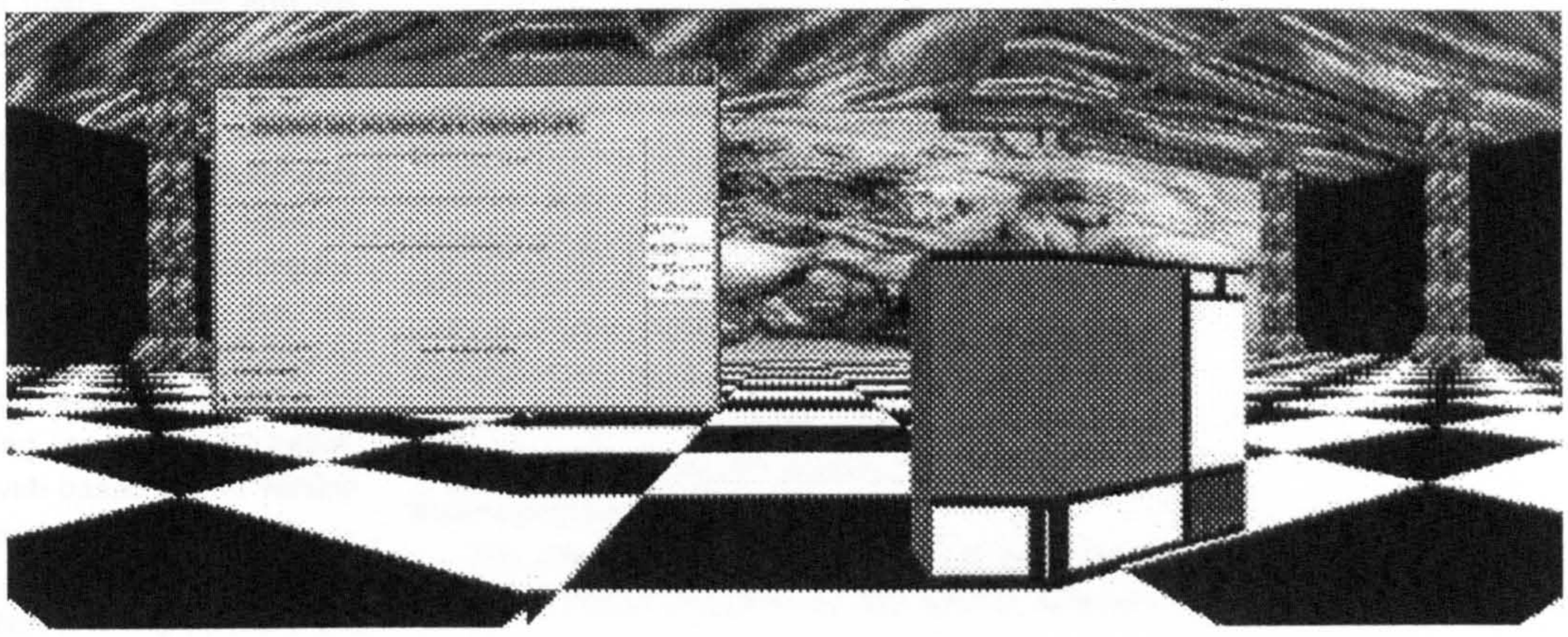


Figure 3. An image of JCAD-VR as it appears when running in the Reality Centre (multi-screen mode)

the virtual world in a database from which it also retrieves information from libraries of objects (**database core**).

As already mentioned the entire framework is based on a client/server architecture.

The services unit is indeed developed across two independently coded packages of the framework - *client* and *server*- and the **network core** is allowing the transmission of data between the two (fig 4).

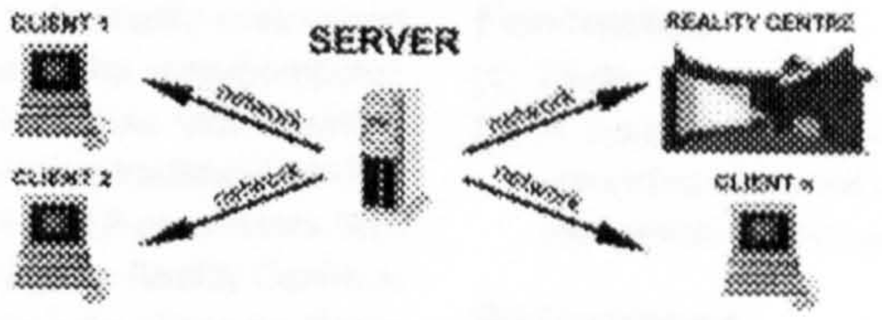


Figure 4. The client/server architecture of JCAD-VR where the server broadcasts to several clients including the Reality Centre.

The **network core** is thus based on a server, several clients and the network allowing the communication. The server is the data-delivering unit that looks after the information to be broadcast. The clients are the users themselves who perform actions and queries, when active, and when passive, rely on the server for receiving data update. The intrinsic multiplatform nature of JCAD-VR, inherited from the language used to code it, allows the server to transmit data to a broad range of machines, from normal PCs to the supercomputer running the Reality Centre, and leaves the research team the freedom to test the software with several operating systems. The communication channel ensures the link between server and clients through a TCP/IP network. As an independent part of the framework the server has an autonomous and simpler interface that provides primarily information about the network system. A number of components are envisaged such as the communication status, the users on line and VR shared environments. Since the clients are communicating through independent processes in the future a further enhancement will allow the server to be capable of dealing with several VR environments simultaneously. At the present stage the network module is supporting use of avatars representing the users inside the virtual world, and it gives also the possibility to interact between the users through a chat system and a whiteboard for sketching. Support for voice communication is being considered for further development as well as a web based applet version of the client side of the software.

The **database core** includes an internal database, that keeps track of the creation or manipulation of the objects in the virtual scene, and an external database through which users will be able to retrieve geometric primitives, materials, lights etc. The internal database is closely coupled with the **network core**. It is not only keeping track of what is happening in the user's virtual world but, most importantly, it receives, through the network, information sent by other users' internal databases. If a new object is created or its status is changed the system will upgrade the internal database of each user no matter who is doing the action. For the convenience of the user an I/O module will allow import/export of objects from/to other packages.

Technical overview

The entire system is coded in Java'. The choice, even if less efficient in term of performances if compared with some other languages, offered indeed great flexibility, true scalability and last but not least fully multi-platform support. Moreover the use of Java' programming language became a natural choice when its 3D suite was released (Java3D'). Its network-centric nature, its multimedia integration together with the use of native hardware acceleration (OpenGL) and multi-processors support (in the case of Sgi workstation) makes it the perfect choice for the development of a real-time multimedia collaborative system. Furthermore thanks to Java''s performance scalability and hardware independence the concept of CAAD has been pushed even further creating a VR environment that can co-exist between high-end supercomputers and common PCs.

The client application, in response to the obvious hardware limits imposed by the use of different

hardware, has been written to be easily customised to run on PCs as well as on a Sgi supercomputer. The former are normal PCs whose video-card is displaying the virtual world only on a traditional window or at full screen. The latter is a 12-processors 6Gb Ram Sgi Onyx2 system running the Reality Centre at ABACUS, University of Strathclyde, Glasgow. When the JCAD-VR is launched on the Sgi it can take advantage of its computational power to stretch itself on a 5 metre wide 2 metre high tassellated screen where 3 Barco projectors create a 160 degree panoramic image.

Conclusions and further developments

The multidisciplinarity of this research is giving the opportunity to investigate collaborative design issues, the role of interfaces inside CAAD packages, the design process in the first stage of its conception, the use of Virtual Reality in architecture and last but non least, a number of technical issues.

As already pointed out, JCAD-VR is an ongoing project and several enhancements are planned for the next releases aiming to get that feeling of intuitiveness and that control over the design that should be the goal of every application using Virtual Reality.

Footnotes

[1] Dorta, T. and LaLande, P., 1998, p.144-148.

[2] A tessellated screen is a composite screen consisting of several projectors creating an unique high resolution image.

References

Ucelli, G., Conti, G., Lindsay, M. and Ryder, G.: 2000, from "Soft" to "Hard" Prototyping: A Unique Combination of VR and RP for Design, in proceedings from UkVRSig 2000, Glasgow.

Bertol, D.: 1997, Designing Digital Spaces, An architect's guide to Virtual Reality, John Wiley & Sons, Inc.

Dorta, T. and LaLande, P.: 1998, The impact of Virtual Reality on the design process, in proceedings from ACADIA Conference, Quebec City.

Hughes, M., Hughes, C., Shoffner, M. and Winslow, M.: 1997, Java Network Programming, Manning, Greenwich.

Barrilleaux, J.: 2000, 3D User Interfaces with Java 3D, a guide to computer-human interaction in three dimensions, Manning, Greenwich.

Sowizral, H. A. and Deering, M. F.: 1999, The Java 3D API and Virtual Reality, IEEE Computer Graphics and Applications, May/June.

Ucelli, G., Conti, G., Petric, J. and Maver, T. (2002). Real Experiences of Virtual Worlds. In D. Marjanovic (Ed.), *Proceedings of the Design 2002, 7th International Design Conference, 2002, Cavtat* (pp. 561-566). Zagreb, Croatia: Faculty of Mechanical Engineering and Naval Architecture, Zagreb and The Design Society, Glasgow, UK.

INTERNATIONAL DESIGN CONFERENCE - DESIGN 2002
Dubrovnik, May 14 - 17, 2002.

REAL EXPERIENCES OF VIRTUAL WORLDS

G. Ucelli, G. Conti, J. Petric and T.W. Maver

Keywords: Virtual Reality, Collaborative Design, Distributed Environment

1. Introduction

The present use of 3D simulations or more effective virtual worlds has provided the designer with new media capable of storing several levels of information traditionally obtained only with the help of multiple media, usually more time and resource-consuming.
Virtual models in particular can store information about planning issues, geometric design, material choices or even furniture and lighting conditions. This level of representation provides the designer with all the necessary tools to represent an architectural environment and facilitate the research of potentially good design solutions.
The use of Virtual Reality (VR) within the design process has not only enabled the designer to store more information than with the use of the traditional media and to check the design solutions more efficiently but furthermore it has enhanced the level of simulation providing:

- **Immersion**: Users are completely surrounded by the environment.
- **Presence**: Being surrounded the participant has actually the sensation of being *in* the environment. The Virtual Environment becomes then a place on its own and its perception is similar to real environments.
- **Interactivity**: This is surely the most important *feature* provided by VR: the environment allows the participant to be involved and the result of the actions done by the participant is visualized in the VE.
- **Autonomy**: Participants are neither constrained in paths nor in views preset by others but have the freedom and autonomy to explore any single part of the environment.
- **Collaboration**: Multiple users are able to take part and to interact in the same VE.

The use of VR can also broaden the boundaries of traditional perception to give the experience of worlds not necessarily real or material and to give the freedom to safely simulate dangerous or expensive condition for training purposes. In fact some applications can simulate something completely different from anything we have ever directly experienced such as the visualisation of the ebb and flow of the world's financial markets or the information of a large corporate database. Other applications provide ways of viewing from an advantageous perspective not possible or too expensive in the real world, like scientific simulators, tele-presense systems and air traffic control systems.
The speed at which technology is evolving is making the application of VR within the design professions a feasible approach. AEC companies have already started to evaluate how time consuming the traditional presentation path can be where animations or walkthroughs are used to show designs solutions to their clients. In fact traditional CAD/CAAD systems are used as rendering tools more than design tools. Any change on design solutions is subject to the inevitable delay of having to step back to the CAD/CAAD systems and then the result must be rendered again to be eventually visualized. This approach is obviously not only unconvenient but time consuming and therefore costly. The consequence of these issues is that some design and manufacturing companies have already started to

investigate how VR can be used within the design process.

2. Background

Since the presentation of The CAVE at the SIGGRAPH conference in 1992 all the media have been presenting VR technologies as the new tools that designers were waiting for and in particular they have been erroneously assumed that it would have triggered the architects' interest for its power to communicate design ideas.
Unfortunately VR is far from being used on a large scale during the architectural design activities although it would provide a natural and easy-to-understand interface between practitioners and clients. Moreover VR would enable architects to test the design functionality and to see whether the design solutions reach the clients expectations; it would increase the possibilities to design a better-built environment by:

- addressing sustainability through environmental simulations and appraisal
- engaging design creativity through immersive design

We have identified two important reasons why VR has not been widely used in the architectural context: the lack of interfaces designed for architects and the wrong positioning of the VR phase inside the design process.
If we observe the software available on the market from a user-centric point of view it is clear that none of the current packages used by architects provides an easy interface to generate virtual environments during the creation phase of the design process. CAAD packages are often complex rendering tools more than design tools and the funtionalities related to VR they provide are subordinated to the creation of 3D models and exportation. Consequently any change on design solutions is subject to the inevitable delay of going back to the CAAD system to refine the 3D model. Thus the recreation of 3D models is so impractical and time-consuming that becomes worth doing only when every design decision has been already taken. In these circumstances the use of VR would just increase costs.
Furthermore the lack of interfaces suitable for architects has as a consequence the wrong positioning of VR technologies in the design process. In fact if the virtual environments created using CAAD packages are generated as refinements, adjustments and exportation from traditional 3D scenes, it is clear that practitioners will consider their use only as presentation tools. Keeping up to date 3D models is an expensive task and obviously even more expensive is to upgrade VEs generated from them.
Therefore we could say that VR is relegated to the end of the design process rather then being used to engage design creativity through immersive design.

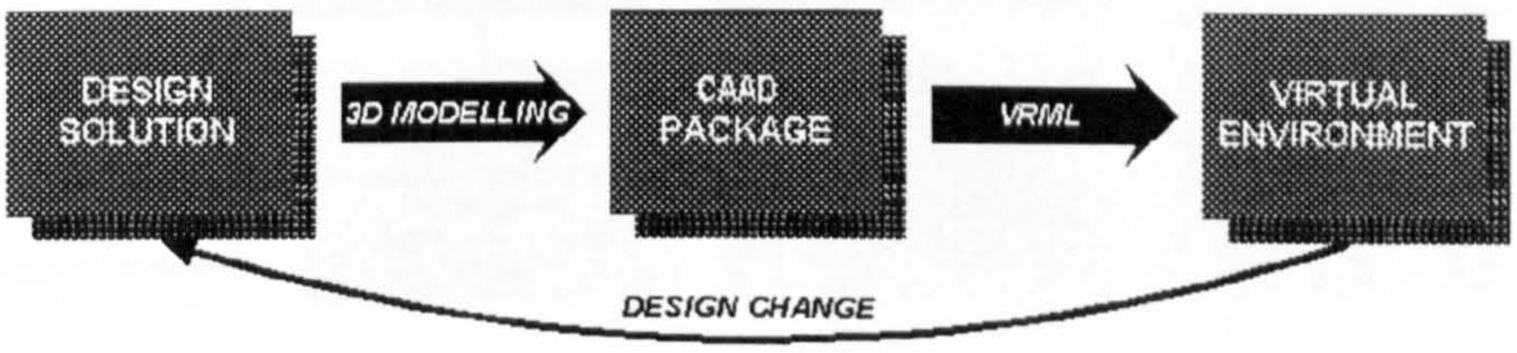


Figure 1. Traditional Schema

Having identified these problems behind the difficulty of using VR from the very beginning of the design phase, the research group has thought to engage itself in the development of a VR system that provides a flexible user-friendly immersive environment to support collaborative design on a synchronous co-editing base, being called JCAD-VR.
This paper will report the present state of the JCAD-VR system and will highlight its future development.

3. The JCAD-VR schema

The idea upon which the JCAD-VR framework is founded is to anticipate the use of VR within the creation phase thus taking full advantage of VR technology. The system in fact allows the creation of simple virtual environments through a user-friendly interface without forcing the user to model it with

traditional CAAD packages. The use of CAAD packages is therefore left to the final stage of the project, where further refinements are needed.
It creates simple parametric 3D-shapes directly in a co-edit VR environment, thus allowing the design to be shared as it evolves.

Figure 2. JCAD-VR Schema

To allow constant collaboration between several users the entire project is based on client-server architecture where every user accesses the virtual world, interacts with the VE and shares design tasks. The whole framework is organised in an object-oriented fashion, where each module fulfils a certain task and it is independently coded. This approach has allowed the delivery of an initial functioning core of the system, whose capabilities will be expanded in the near future.

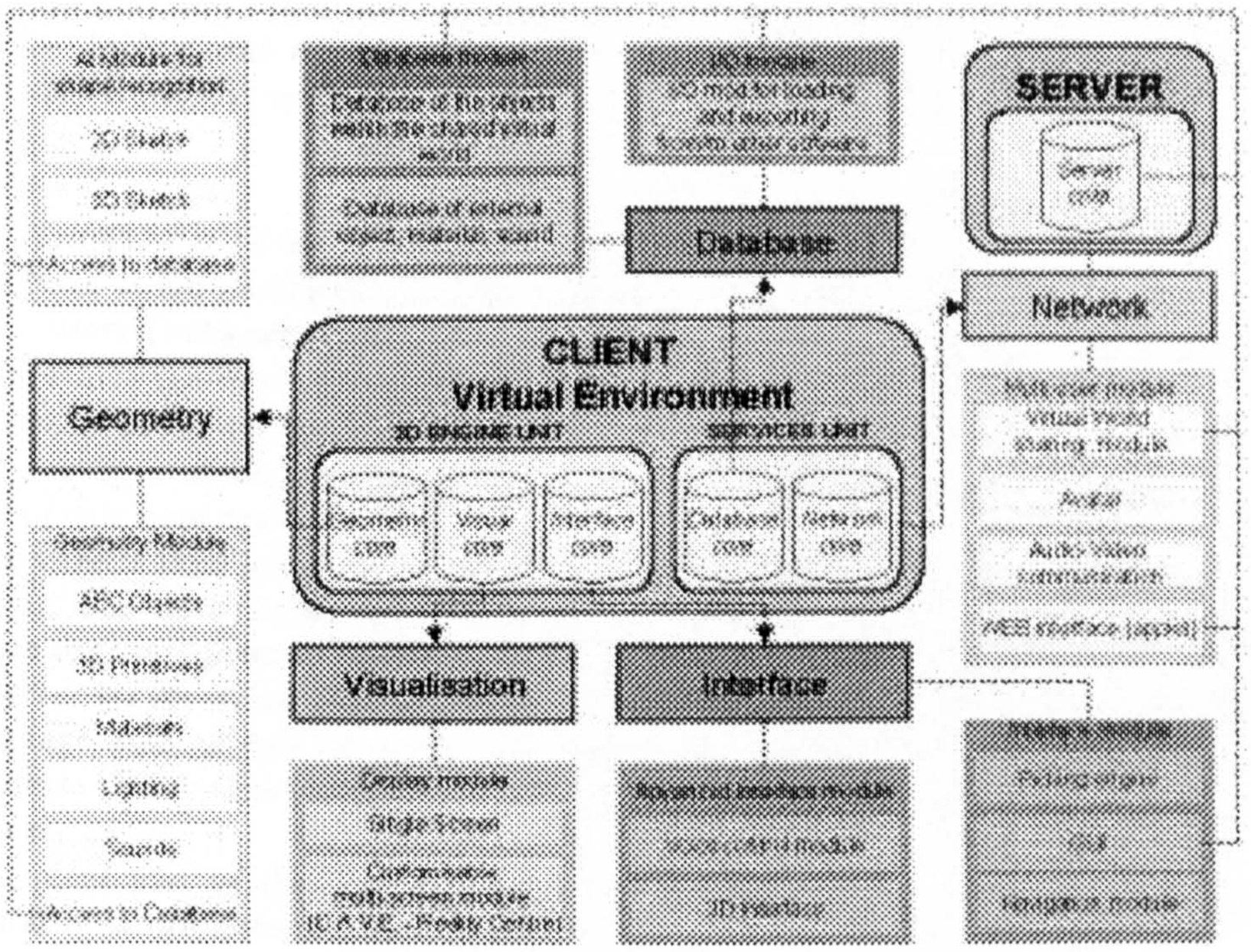


Figure 3. The JCAD-VR framework schema

From the implementation point of view JCAD-VR handles the VE through two closely connected sections: a *3D engine* and a *services unit* each made of several *modules*.

3.1 3D engine unit

The 3D engine handles all the information regarding the visual aspects of the VE. It includes the code necessary to create and modify geometric entities (**geometry core**), to run the 3D-interface (**interface core**) and to deal with several different output devices (**visual core**).
The first module of the **geometry core** handles the creation of 3D objects: both geometric primitives (cones, boxes, spheres etc.) and architectural entities (walls, slabs etc.). To the architectural entities some extra properties were provided such as: information on internal and external faces or windows and doors attached to them.
The *geometry module* will also provide the means for attaching materials to objects and add lights and

objects from a library to the virtual world through the *database module*.
The **interface core** does not implement a traditional graphic user interface (GUI): JCAD-VR has been provided with a 3D interface that is an integral part of the virtual world itself. The idea behind it is that instead of the traditional menus and toolbars the UI is immersed in VE providing the means for the interaction: 3D menus pop up showing 3D icons and the 3D menus themselves can be moved for the convenience of the user. Visual feedback is provided in the form of rulers showing the size of objects or 3D icons helping the user in the operations to be done on the objects.
The **visual core** is the part of the framework that allows the interfacing with the visualization devices. The client application has been implemented in order to be used on PCs as well as on Sgi supercomputers. The former are normal PCs whose video-card is displaying the virtual world only on a traditional window at full screen, the latter is a 12-processors 6Gb Ram Sgi Onyx2 system running a Reality Centre. When JCAD-VR is launched on the system running the Reality Centre it can take advantage of the increased computational power stretching its visual output on a 5 metre wide 2 metre high tassellated screen where 3 projectors create a 160 degree panoramic image.
For the sake of flexibility the entire system is coded in Java™. The choice, even if less efficient in terms of performances if compared with some other languages, offered indeed great flexibility, true scalability and last but not least fully multi-platform support. Moreover the use of Java™ programming language became a natural choice when its 3D suite was released (Java3D™).
This choice has provided the flexibility neceessary to deliver images for a range of viewing devices and the internal architecture of the visual core is such that modules might be easily adapted to allow use of different VR devices such as CAVEs or Headmounted Displays.

3.2 Services unit

The *services unit* handles all the circulation of data within the system.
It is the backbone of the interconnection between users: it manages network connections, it exchanges data between users (**network core**) and it keeps track of the state of the virtual world through a database from which it also retrieves objects information (**database core**).
The *services unit* is based on a client/server architecture therefore it is implemented across two independent packages of the framework the *client* and the *server* and the **network core** allows the transmission of data between them.
The **network core** is thus based on a multi-client server, several clients and the network allowing the communication. The server is the data-delivering unit that looks after the information to be broadcast. The clients are the users themselves who perform actions and queries, when active, and when passive, rely on the server for receiving data update. The intrinsic multiplatform nature of JCAD-VR, inherited from the language used, allows the server to transmit data to a broad range of machines across several operating systems. The communication channel ensures the link between server and clients through a TCP/IP network.
As an independent part of the framework the server has a simple and autonomous interface that provides primarily information about the network system.
At the present stage the *network module* supports:

- Broadcasting of new geometries in the VE
- Notification of creation of new geometries in every user's internal database and broadcasting of their numerical information
- Broadcasting of modifications applied on geometries in the VR scene
- Notification of changes on geometries in every user internal database
- Checking for user priority on the objects through a distributed locking mechanism
- *Avatars* representing multiple clients in the VE
- Interaction between users through a chat system and a whiteboard for freehand sketching in 2D.

It will be soon expanded to include new functionalities such as the transfer of voice and video accross users.

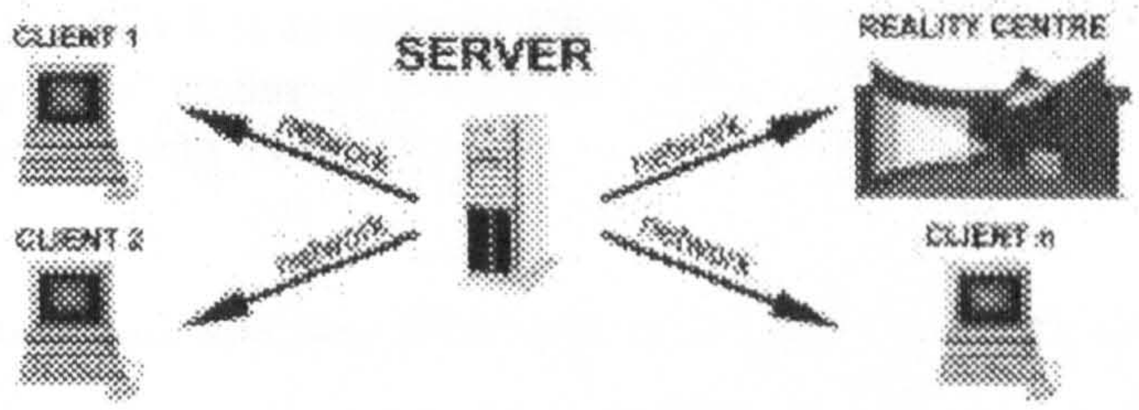
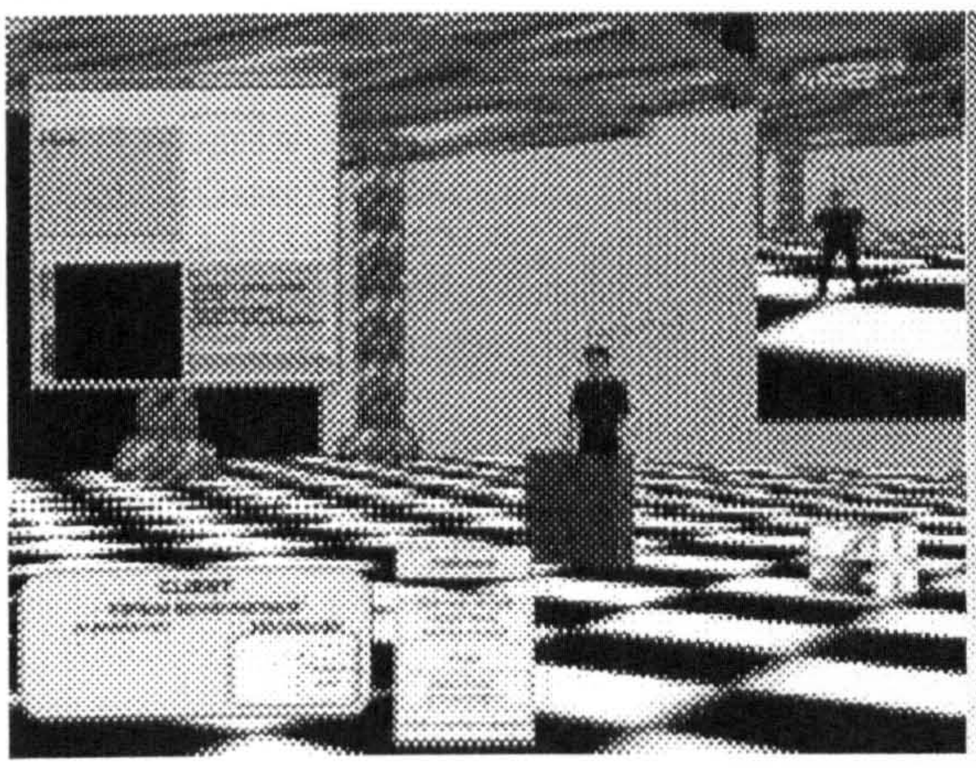


Figure 4. The client/server architecture of JCAD-VR where the server broadcasts to several clients including the Reality Centre

The **database core** includes an internal database, that keeps track of the numerical parameters of the geometries created or modified within the virtual scene, and an external database through which users will be able to retrieve more complex 3D shapes, AEC objects, materials, lights etc.
The internal database is closely coupled with the **network core**. Not only it keeps track of what it is happening in the user's virtual world but also, most importantly, it receives, through the network, information sent by other users' internal databases. If a new object is created or its geometric parameters are changed the system will upgrade the internal database of each user no matter who is doing the action.
For the convenience of the user an *I/O module* supports loading of external files thus allowing import from traditional CAAD packages.



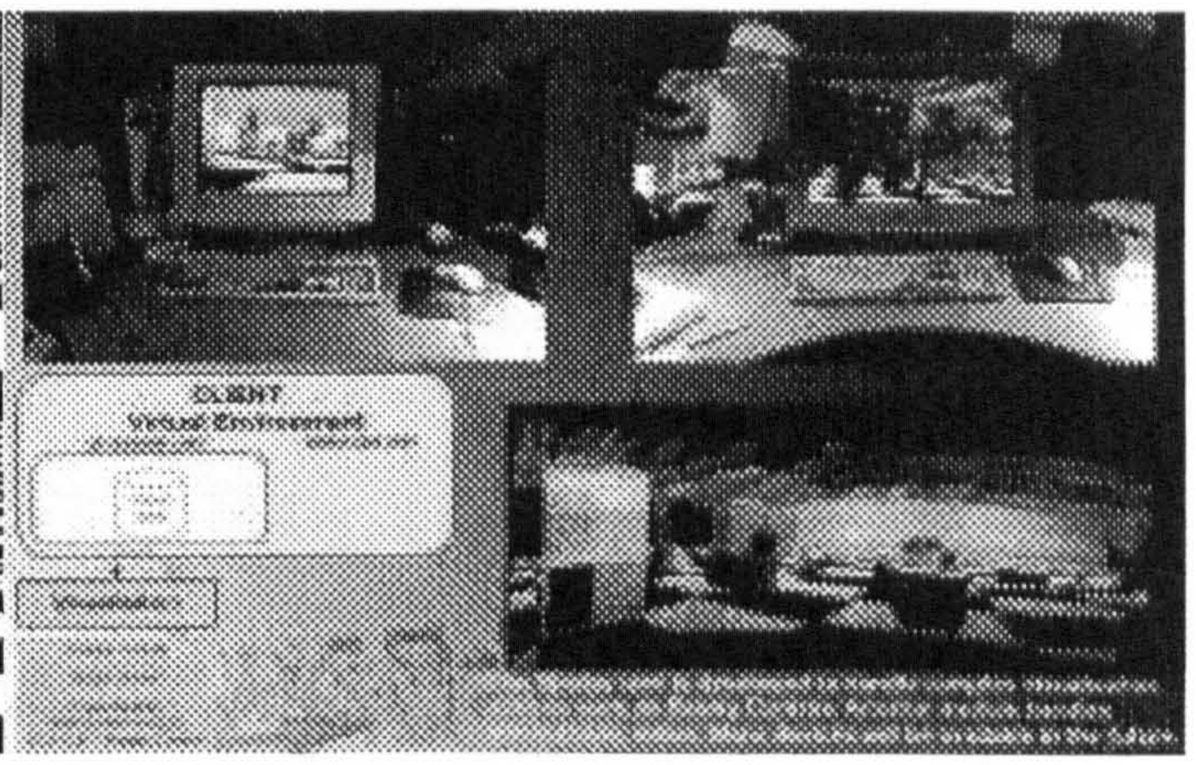


Figure 5. Avatars and communication tools in JCAD-VR. Figure 6. JCAD-VR running on a PC, Sgi and in the Reality Centre

4. Further Developments

Further developments and completion of several modules of the framework are planned to be carried out in the near future:

- A voice driven interface enhancing user friendliness of the UI
- Implementation of the Audio/Video module and introduction of Video Conferencing facilities.
- Support for driving devices such as 6-degree of freedom/virtual glove
- Implementation of a multi-environments server capable of dealing with several VR environments simultaneously.

5. Conclusions

The JCAD-VR framework has confirmed the possibility of anticipating the use of VR within the creation phase thus taking full advantage of VR technology. It allows access to VR in an easy and direct way not obliging users to model in traditional CAAD environments.
Moreover it creates simple parametric 3D-shapes directly in a co-edit VR environment, thus allowing the design to be shared as it evolves.

As already pointed out, JCAD-VR is an ongoing project and several enhancements are planned for the next releases aiming to get that feeling of intuitiveness and that control over the design that should be the goal of every application using Virtual Reality.

References

Barrilleaux, J.,"3D User Interfaces with Java 3D-A guide to computer-human interaction in three dimensions", Manning, Greenwich, 2000.

Bertol, D., "Designing Digital Spaces", John Wiley & Sons, 1997.

Conti, G., Ucelli, G. and Maver, T., "JCAD-VR - A Collaborative Design Tool in Virtual Reality", Proceedings of ECAADE 19. Helsinki, 2001, pp 454 – 459.

Dorta, T. and LaLande, P., "The Impact of VR on the Design Process", Proceedings of ACADIA '98, Quebec, 1998, pp 139 – 161.

Hughes, M., Hughes, C., Shoffner, M. and Winslow, M., " Java Network Programming", Manning, Greenwich, 1997.

Maver, T., Harrison, C. and Grant, M., "Virtual Environments for Special needs", Proceedings of CAAD Futures 2001. Eindhoven, 2001, pp 151 – 159.

Petric, J., Ucelli, G. and Conti, G., "Educating the Virtual Architect", Proceedings of ECAADE 19. Helsinki, 2001, pp 388 – 393.

G. Ucelli
ABACUS Unit,
Architecture and Building Science,
University of Strathclyde,
131 Rottenrow
G4 0NG Glasgow, UK
Tel. +44 141 548 3997
Fax. +44 141 552 3996
Email: giuliana.ucelli@strath.ac.uk

Petric, J., Maver, T., Conti, G. and Ucelli, G. (2002). Virtual Reality in the Service of User Participation in Architecture. In K. Agger, P. Christiansson and R. Howard (Eds.), ***Distributing ·Knowledge in Building. Proceedings of CIB W78 Conference, Aarhus*** **(pp. 217-224). Aarhus, Denmark: Aarhus School of Architecture and Centre for Integrated Design.**

Theme: Arial, size 10 (will be completed by the organising committee)

Title: **Virtual Reality in the Service of User Participation**

Author(s): J Petric, T Maver, G Conti, G Ucelli
Institution(s): University of Strathclyde
E-mail(s): j.petric@strath.ac.uk
t.w.maver@strath.ac.uk
giuseppe.conti@strath.ac.uk
gucelli@strath.ac.uk

Abstract:

The issue of user participation in the processes of building and urban design is enjoying renewed attention following its relative neglect over the last 20 years due, in large measure, to significant advances in emerging information technologies, particularly multimedia, virtual reality and internet technologies.

This paper re-established the theoretical framework for participatory design evolved in the late sixties and early seventies as part of the movement towards a more explicit design methodology and attempts an explanation of why the concept failed to gain commitment from the architectural and urban design professionals.

The paper then gives an account of two significant developments in the evolution of the application of information technologies with which the authors have been engaged. These are:

i. a responsive and interactive interface to wholly immersive and realistic virtual reality representations of proposed buildings and urban neighbourhoods.

ii. an intuitive and platform-independent VR modelling environment allowing collaborative evolution of the scheme from within the virtual world.

The impact of these IT developments is demonstrated in the context of the design of a leisure facility for a community of users with physical impairment.

Keywords: *user, design, participation, VR, CAAD*

Design Decision Making

Architectural design is a multi-faceted occupation which requires, for its successful performance, a mixture of intuition, craft skills and detailed knowledge of a wide range of practical and theoretical matters. It is a cyclical process in which groups of people work towards a somewhat ill-defined goal in a series of successive approximations. There is no 'correct' method of designing and, although it is recognised that the process can be divided into separate phases, there is no generally accepted sequence of work that might guide design teams in the direction of achieving a satisfactory solution. Indeed, there are no solutions to design problems in the way that there are solutions to mathematical problems: the best that can be hoped for is an outcome which satisfies the maximum number of constraints which bound the area of concern. Furthermore, design is not an algorithmic process in which the desired conclusion can be reached by the application of step-by-step procedures - first finalising this aspect, then that. It is a fluid, holistic process wherein at any stage all the major parts have to be manipulated at once. In this sense, it is less like solving a logical puzzle and more like riding a bicycle, blindfold, whilst juggling.

Despite the complexity of the design decision-making process the emerging new generation of computer-based models is already having an impact on how design is performed and, hence, on the quality of design. The impact stems from the fact that the new models, as opposed to paper-based plans and elevations or other conventional forms, are predictive rather than descriptive; dynamic rather than static; explicit rather than implicit and, above all, permit a more-or-less continuous and interactive assessment of the effects of a developing design on cost and performance.

Evidence is growing of the advantages offered by the application of computers in design, and these can be summarised as follows:

Widening the Search for Solutions

Access to programs which dynamically predict the cost and performances characteristics of optional design proposals can increase the scope of search for good solutions by as much as ten-fold. Not only is the search coverage extended, it is also more purposefully directed because designers are able to compare the quality of any one tentative solution against the quality of all previous solutions.

Greater Integration in Decision-Making

In conventional working, a great deal of design time is lost as proposals are passed to and fro between the architect (who tends to be the originator) and the other specialist members of the design team (who tend to the "checkers"). Quite frequently the scheme on which the architect has lavished time and effort is found by one or other of the specialists to be infeasible. With access to appropriate appraisal techniques embodied in computer programs, it is possible to check a proposal against a wide range of criteria from the outset of the design activity. Moreover, it is entirely practical (though not yet a widespread working method) for all members of the design team to have access to, and operate on, the common design model whether or not they share a design office. The models, then, can provide a strong integrating force in design team working.

Improving Design Insights

Apart from the use of appraisal programs to search for better designs, the programs can be used in a research and development context to provide insights into the way in which particular design decisions affect cost and performance. Typically, a designer working in this mode would select an existing building for study, then, keeping all other design variables constant (insofar as this is possible), systematically vary one factor while recording the cost/performance output from the program. In this manner, the architect can establish sets of causal relationships which provide powerful insights into structure of design decision-making.

Differentiation of Objective and Subjective Judgements

Contrary to the early fears of many architectural practitioners, the use of CAAD techniques focuses increased attention on subjective value judgements rather than less. As measurable attributes of optional designs are made more explicit, the necessary value judgements are forced to the surface of design activity and thereby, themselves become more explicit. The effect of this is to make it clear to designers and their clients, which judgements are based on quantifiable criteria and which on subjective and intuitive concepts.

Evidence of the degree to which computer-generated cost/performance information promotes effective value judgement, throws into sharp focus the crucial question: **whose** value judgement? This question was, for the first time, seriously addressed in the Design Participation Conference in Manchester in 1971 (1). At that time, however, the human-machine interface was too primitive for the concept of useful participation by the users of buildings to be achieved. The new technologies of VR and Multimedia give real prospects for participation.

Virtual Reality

The present use of 3D simulations or more effective virtual worlds has provided the designer and user participants with new media capable of storing several levels of information traditionally obtained only with the help of multiple media, usually more time and resource-consuming.
Virtual models in particular can store information about planning issues, geometric design, material choices or even furniture and lighting conditions. This level of representation provides the designer with all the necessary tools to represent an architectural environment and facilitate the research of potentially good design solutions.
The use of Virtual Reality (VR) within the design process has not only enabled the designer to store more information than with the use of the traditional media and to check the design solutions more efficiently but furthermore it has enhanced the level of simulation providing:

?? **Immersion**: Users are completely surrounded by the environment.
?? **Presence**: Being surrounded the participant has actually the sensation of being *in* the environment. The Virtual Environment becomes then a place on its own and its perception is similar to real environments.
?? **Interactivity**: This is surely the most important *feature* provided by VR: the environment allows the participant to be involved and the result of the actions done by the participant is visualized in the VE.
?? **Autonomy**: Participants are neither constrained in paths nor in views preset by others but have the freedom and autonomy to explore any single part of the environment.
?? **Collaboration**: Multiple users are able to take part and to interact in the same VE.

The use of VR can also broaden the boundaries of traditional perception to give the experience of worlds not necessarily real or material and to give the freedom to safely simulate dangerous or expensive condition for training purposes. In fact some applications can simulate something completely different from anything we have ever directly experienced such as the visualisation of the ebb and flow of the world's financial markets or the information of a large corporate database. Other applications provide ways of viewing from an advantageous perspective not possible or too expensive in the real world, like scientific simulators, tele-presence systems and air traffic control systems.

The speed at which technology is evolving is making the application of VR within the design professions a feasible approach. AEC companies have already started to evaluate how time consuming the traditional presentation path can be where animations or walkthroughs are used to show designs solutions to their clients. In fact traditional CAD/CAAD systems are used as rendering tools more than design tools. Any change on design solutions is subject to the inevitable delay of having to step back to the CAD/CAAD systems and then the result must be rendered again to be eventually visualized. This approach is obviously not only inconvenient but time consuming and therefore costly. The consequence of these issues is that some design and manufacturing companies have already started to investigate how VR can be used within the design process.

The JCAD-VR Prototype

In the Department of Architecture and Building Science at the University of Strathclyde, the ABACUS group has been building a prototype design decision support system known as JCAD-VR (2).

The idea upon which the JCAD-VR framework is founded is to anticipate the use of VR within the creation phase thus taking full advantage of VR technology. The system in fact allows the creation of simple virtual environments through a user-friendly interface without forcing the user to model it with traditional CAAD packages. The use of CAAD packages is therefore left to the final stage of the project, where further refinements are needed.
It creates simple parametric 3D-shapes directly in a co-edit VR environment, thus allowing the design to be shared as it evolves.

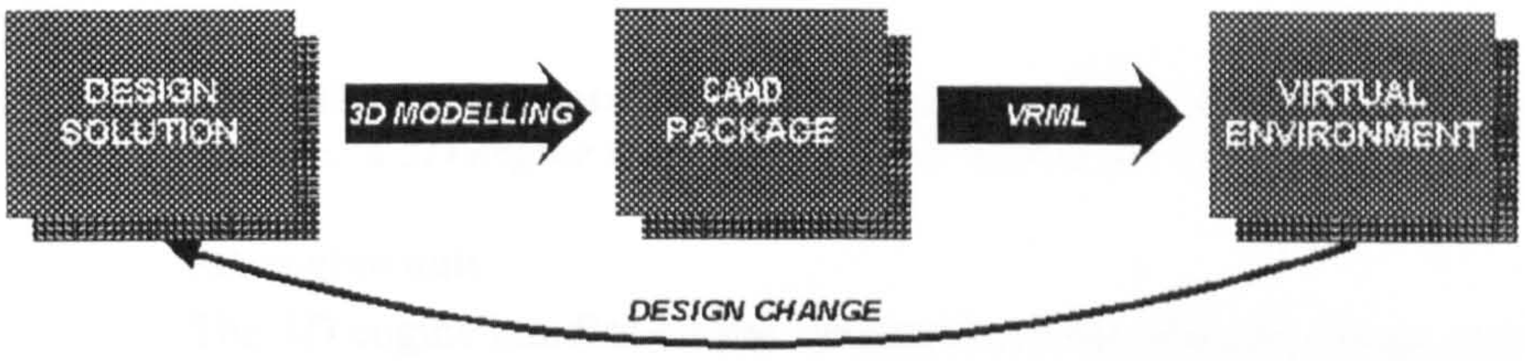


Figure 1. Traditional Schema.

Figure 2. JCAD-VR Schema.

To allow constant collaboration between several users the entire project is based on client-server architecture where every user accesses the virtual world, interacts with the VE and shares design tasks. The whole framework is organised in an object-oriented fashion, where each module fulfils a certain task and it is independently coded. This approach has allowed the delivery of an initial functioning core of the system, whose capabilities will be expanded in the near future.

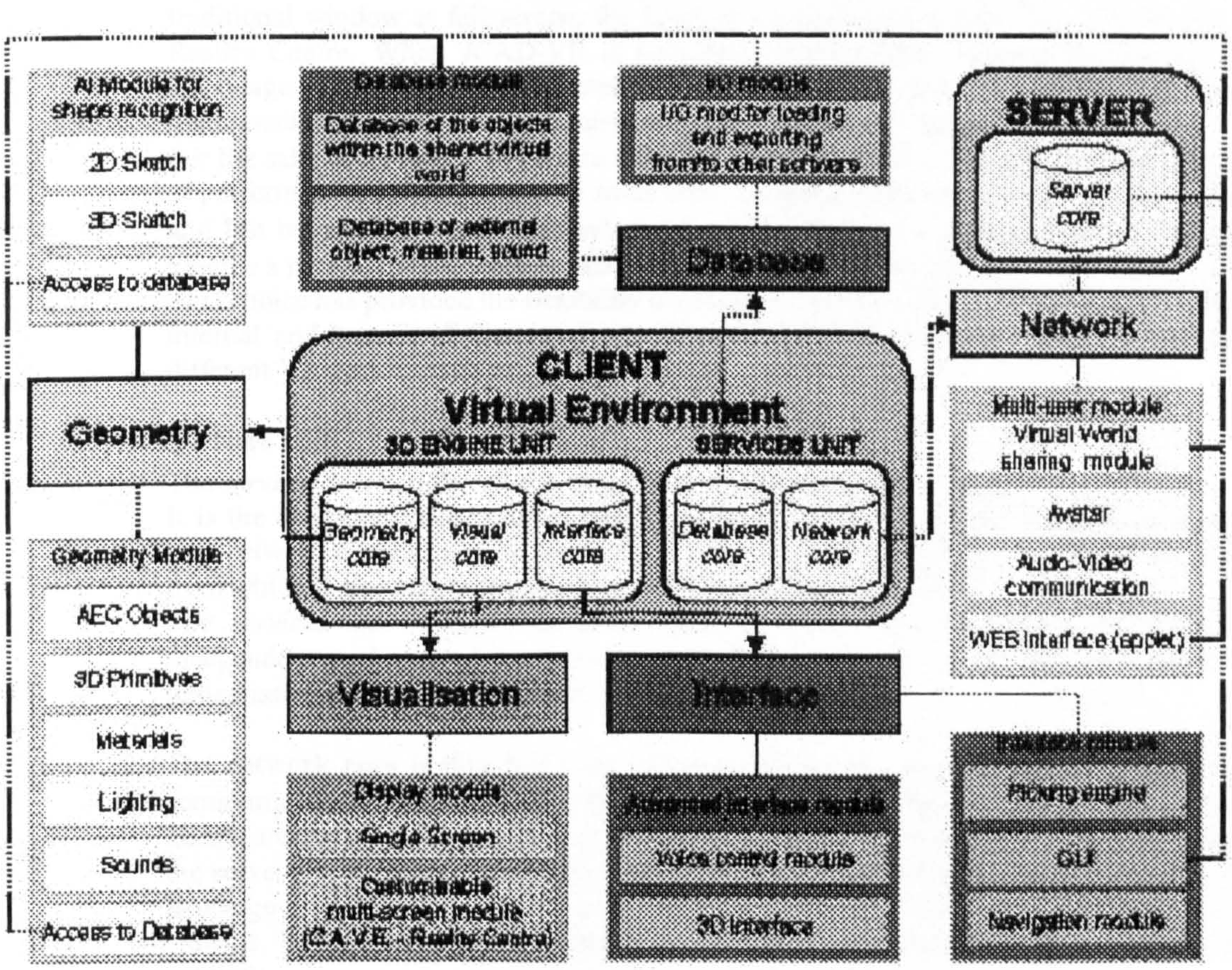


Figure 3. The JCAD-VR framework schema

From the implementation point of view JCAD-VR handles the VE through two closely connected sections: a *3D engine* and a *services unit* each made of several *modules.*

3D engine unit

The 3D engine handles all the information regarding the visual aspects of the VE. It includes the code necessary to create and modify geometric entities (**geometry core**), to run the 3D-interface (**interface core**) and to deal with several different output devices (**visual core**).

The first module of the **geometry core** handles the creation of 3D objects: both geometric primitives (cones, boxes, spheres etc.) and architectural entities (walls, slabs etc.). To the architectural entities some extra properties were provided such as: information on internal and external faces or windows and doors attached to them.
The *geometry module* will also provide the means for attaching materials to objects and add lights and objects from a library to the virtual world through the *database module.*

The **interface core** does not implement a traditional graphic user interface (GUI): JCAD-VR has been provided with a 3D interface that is an integral part of the virtual world itself. The idea behind it is that instead of the traditional menus and toolbars the UI is immersed in VE providing the means for the interaction: 3D menus pop up showing 3D icons and the 3D menus themselves can be moved for the convenience of the user. Visual feedback is provided in the form of rulers showing the size of objects or 3D icons helping the user in the operations to be done on the objects.

The **visual core** is the part of the framework that allows the interfacing with the visualization devices. The client application has been implemented in order to be used on PCs as well as on SGI supercomputers. The former are normal PCs whose video-card is displaying the virtual world only on a traditional window at full screen, the latter is a 12-processors 6Gb Ram SGI Onyx2 system running a Reality Centre. When JCAD-VR is launched on the system running the Reality Centre it can take advantage of the increased computational power stretching its visual output on a 5 metre wide 2 metre high tassellated screen where 3 projectors create a 160 degree panoramic image.
For the sake of flexibility the entire system is coded in Java? . The choice, even if less efficient in terms of performances if compared with some other languages, offered indeed great flexibility, true scalability and last but not least fully multi-platform support. Moreover the use of Java? programming language became a natural choice when its 3D suite was released (Java3D?).
This choice has provided the flexibility necessary to deliver images for a range of viewing devices and the internal architecture of the visual core is such that modules might be easily adapted to allow use of different VR devices such as CAVEs or Headmounted Displays.

Services unit

The *services unit* handles all the circulation of data within the system.
It is the backbone of the interconnection between users: it manages network connections, it exchanges data between users (**network core**) and it keeps track of the state of the virtual world through a database from which it also retrieves objects information (**database core**).
The *services unit* is based on a client/server architecture therefore it is implemented across two independent packages of the framework the *client* and the *server* and the **network core** allows the transmission of data between them.

The **network core** is thus based on a multi-client server, several clients and the network allowing the communication. The server is the data-delivering unit that looks after the information to be broadcast. The clients are the users themselves who perform actions and queries, when active, and when passive, rely on the server for receiving data update. The intrinsic multiplatform nature of JCAD-VR, inherited from the language used, allows the server to transmit data to a broad range of machines across several operating systems. The communication channel ensures the link between server and clients through a TCP/IP network.

As an independent part of the framework the server has a simple and autonomous interface that provides primarily information about the network system.
At the present stage the *network module* supports:

- ?? Broadcasting of new geometries in the VE
- ?? Notification of creation of new geometries in every user's internal database and broadcasting of their numerical information
- ?? Broadcasting of modifications applied on geometries in the VR scene
- ?? Notification of changes on geometries in every user internal database
- ?? Checking for user priority on the objects through a distributed locking mechanism
- ?? *Avatars* representing multiple clients in the VE
- ?? Interaction between users through a chat system and a whiteboard for freehand sketching in 2D.

It will be soon expanded to include new functionalities such as the transfer of voice and video across users.

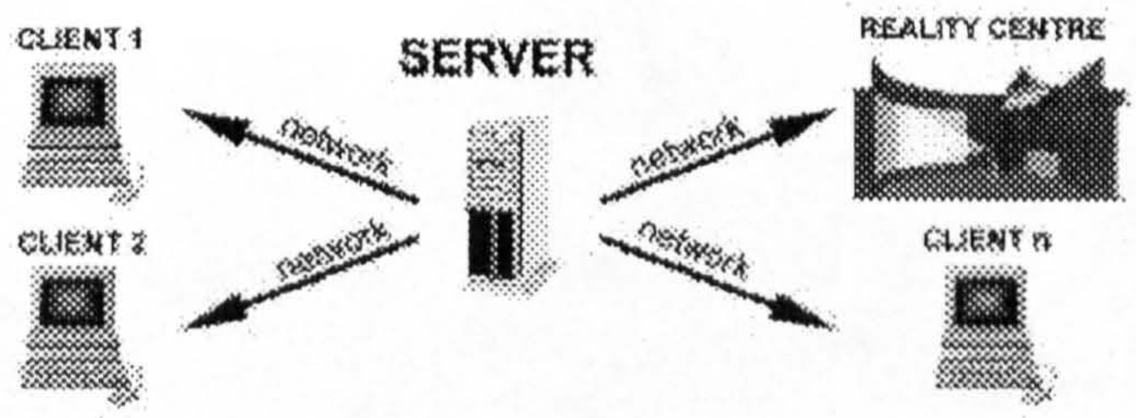


Figure 4. The client/server architecture of JCAD-VR where the server broadcasts to several clients including the Reality Centre

The **database core** includes an internal database, that keeps track of the numerical parameters of the geometries created or modified within the virtual scene, and an external database through which users will be able to retrieve more complex 3D shapes, AEC objects, materials, lights etc.
The internal database is closely coupled with the **network core**. Not only it keeps track of what it is happening in the user's virtual world but also, most importantly, it receives, through the network, information sent by other users' internal databases. If a new object is created or its geometric parameters are changed the system will upgrade the internal database of each user no matter who is doing the action.
For the convenience of the user an *I/O module* supports loading of external files thus allowing import from traditional CAAD packages.

Initial Trials

The current prototype version of JCAD-VR is starting trials of its simultaneous use at the Technical University of Eindhoven and the University of Strathclyde. However, an earlier version was piloted with a group of students in the BSc (Architectural Studies) at the University of Strathclyde.

The brief for the design project associated with the workshop was quite demanding: A Sailing Club for the Disabled located on a canal site in Glasgow City Centre. A real client community agreed to be involved in the project and in the assessment of its outcomes (3).

Within the overall JCAD-VR system, students were encouraged to use initially a standard CAAD package for initial creation of the geometry and then to refine the design using VRML. Event management in their virtual worlds was done through the user of sensors and connectivity. These include touch sensors, proximity sensors, time sensors and anchors.

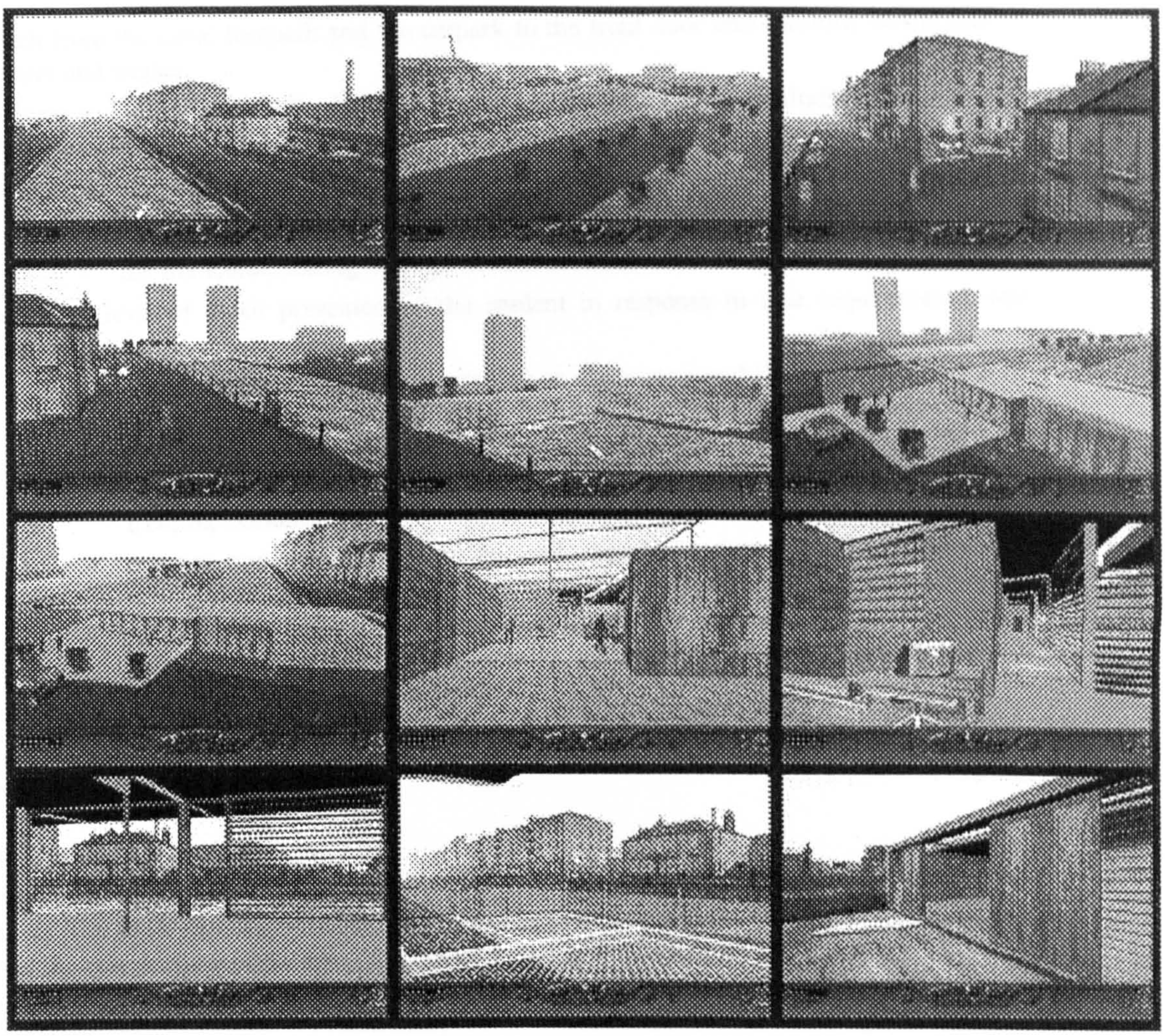


Figure 5. Sailing Club for the Disabled at Spier's Wharf, Glasgow.

The design outcomes

The outcome of the experiment – although not statistically measurable – was nonetheless considered to be remarkable by both the tutors and the client community. One second year student in particular made the most effective use of the full range of functionality of the system.

In relation to the site and exterior of the building these were:

?? Good balance between modelling of 3D geometry and texture mapping to provide a thoroughly convincing, large scale model of the site, which includes existing buildings of importance to the intervention as well as the wider urban issues such as the adjacent motorway.

?? Ability to approach the site, as would a user, by sailing along the canal or, as a wheel chair user, to open gates and wheel along foot-paths to the building entrance.

?? Understanding of the urban site by "flying" nearer or further from the adjacent motorway (along which cars are speeding) to check the attenuation of noise pollution.

In relation to the building itself, the contribution of the student's ability to "visit" his design, during the evolution, as would a wheelchair user, was clearly evident in the quality of the design solution and was manifested in a number of subtle but important ways, for example:

?? The approach from the canal footpath and the carpark to the front door was carefully considered in terms of slopes and angles.

?? The door access and view lines of wheel chair users, on entering the facility (including signage) were completely thought through.

?? The elegance of articulation of the building into two zones - wet and dry - was the evident result of:

- the Boolean operations performed by the student on the volumes;
- the immediate testing of these in the virtual environment.

?? The unparalleled level of detail presented by the student in response to user requirements, was exemplified by :

- the transparency of the balustrades on the ramps and the sophisticated louvre system on the external glazing - a direct result of the designer's perception, from the user viewpoint - of what is important to someone in a wheelchair. Through every window in the building the student was able to show the appropriate view from the building.
- concern and commitment to the real experience for a wheelchair user. The sliding door towards the canal could be opened and the user wheeled out into a mesh deck in order to get a first-hand experience of being "on" the water before perhaps, two ramps down, actually getting into a canoe, the mechanism illustrated by an elegant animation.

The client community group were presented with the work of the students, and were impressed by the sensitivity with which the brief was addressed. They are featuring the outcome on their website (http://www.fcccp.org.uk).

Conclusions and future aspirations

As in the earliest days of the introduction of computers into architectural design, the quantum jump is made by students. The work reported here, and which will be shown during the conference is, we believe, the epitome - in the current state of the art - of excellent practice. It makes a breakthrough, we believe, in the evolution of good design ideas, modelled offline but appraised interactively and offers a real prospect for user participation.

There is some way to go, of course, to *design* interactively in a virtual environment. The next step which we envisage is to link to the 3D model the emerging and sophisticated software for the thermal, lighting and acoustics properties of the building. This would allow the user to visualise, dynamically, airflow, temperature gradients, lighting levels and to experience the actual acoustic characteristics of the space as she/he moves through it.

The other exciting development is for representatives of the client/user group to "join" the designer within the virtual environment and to participate directly in the evolution of the design concept. The recent development of a wheel-chair motion platform for immersive virtual environments (4) will allow the future users of buildings like the Sailing Club for the Disabled to navigate themselves, in their own wheelchair, through the virtual building.

References

1. Nigel Cross, Design Participation. Proceedings of the Design Research Society, Academy Editions (1971)
2. Guiliana Ucelli et al Real Experiences of Virtual Worlds. Design 2002, International Design Conference, Dubrovnic (2002)
3. Jelena Petric et al Educating the Virtual Architect. Promise and Reality. Proceedings of ECAADE Conference, Weimar [Ed Donath, D]
4. Tom Maver et al Virtual Environments for Special Needs. Proceedings of CAAD Futures 2001, (2001)

Petric, J., Ucelli, G. and Conti, G (2002). Real Teaching and Learning through Virtual Reality. In K. Koszewski and S. Wrona (Eds.), *Design e-ducation. Proceedings of the 20th Conference of eCAADe, Warsaw, 2002* (pp. 72-79). Warsaw, Poland: Drukarnia Braci Ostrowskich.

Real Teaching and Learning through Virtual Reality

Jelena Petric, Giuliana Ucelli, Giuseppe Conti
ABACUS, University of Strathclyde, Glasgow, UK
http://www.strath.ac.uk/Departments/Architecture

Abstract. This paper addresses an articulated vision of Virtual Reality which lends itself to design collaboration in teaching and learning and communication of architectural design ideas among students, design professionals and client body during the early stages of the design process.

Virtual Reality (VR) has already acquired a new degree of complexity through development of network-based virtual communities and the use of avatars. The intrinsic quality of VR technology is to support collaborative design experience.

The design tools developed for this experiment are capable of creating 3D objects in a shared VR environment, thus allowing the design and its evolution to be shared. The choice of programming language (Java™) reflects the desire to achieve scalability and hardware independence, which in turn allows for creation of a VR environment that can co-exist between high-end supercomputers and standard PCs. The prototype design environment was tested using PC workstations and an SGI system running a Reality Centre.

The research and teaching/learning experience in the collaborative design environment reported in this paper describe the development and application of software that aims to increase the opportunity for architects to collaborate within virtual worlds which enable effective and transparent information exchange.

Keywords
VR, Collaborative Design, Virtual Environment, Interface, Architecture, Experiment, Design Process.

Introduction

In the recent decades the design profession has been deeply affected by the *digital revolution* and the use of Computer Aided Architectural Design (CAAD) tools is nowadays part of the daily practice in most architectural firms. But in the last few years the 'CAAD community' is experiencing a new revolution that is leading the move from static representation, based on 2D renderings or pre-recorded animations (considered as a sequence of 2D images), to dynamically generated 3D representations. Real-time navigation and interaction, typical of VR environments, provide just that fluent interface enabling the exploration of the design proposals that architects have not been able to get with any other media.

The increasing growth of computational resources and hardware power is probably preparing the anticipated transition to desktop VR applications, making them truly feasible tools for everyday practice. Furthermore, the recent growth of network-based virtual communities has brought a new level of complexity to the notion of virtual spaces, turning the profession of architect into something that might now resembles the one of the *virtual architect.*

Although VR is nowadays a quite mature technology, it is seldom used in architecture throughout the design process, but more often it is merely used as a powerful presentation technique. Design methodologists in the past agreed on the need for iterative cycles between several phases of the design process. From studies concerning designers' behaviour [6] many authors observed an indefinite number of return loops from the moment when gathering of information and structuring of the design problem take place (known as *analysis*) to the one when design solutions are generated (known as *synthesis*).

The use of Virtual Reality within the design process provides the designer with an appropriate, quick and practical feedback which facilitates search for design solutions. Moreover, it enables the capture of more information than would be possible to capture with the use of the traditional media and makes the checking of the design solutions more efficient by enhancing simulation capabilities. The use of VR in design broadens the boundaries of traditional perception by

providing experiences of worlds not necessarily real or material. In short VR is the perfect simulation medium for architects investigating design solutions.

It is then highly predictable that in the near future VR will become the interface for the next generation of computer aided design (CAD) applications and we can anticipate the change of its use from a mere presentation medium to a more powerful and effective design tool.

At present Virtual Environments (VE) are often created using CAAD packages for refinements, adjustments and exportation based on traditional 3D scenes. Documenting the evolution and development of design by constant updating of 3D models is an expensive business, and obviously even more expensive is the upgrading of the VEs generated from these models. In most existing design scenarios the decision making process does not take advantage of the technology, but relegates the use of VR to the end of the process as a more convincing tool to impress contractors and clients. In such a scenario, only once the final solution has been achieved is it worth investing time into more powerful visualisation media.

With this background knowledge the research task we set ourselves was to develop the VR system which would help designers in the initial stages of the design process to take advantage of the VR as a new design tool.

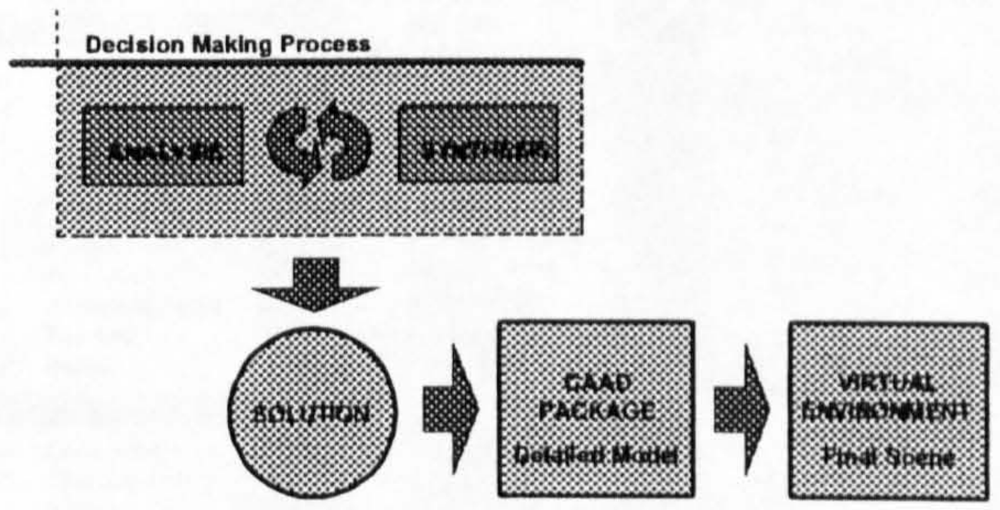


Figure 1. The current position of VR within the design

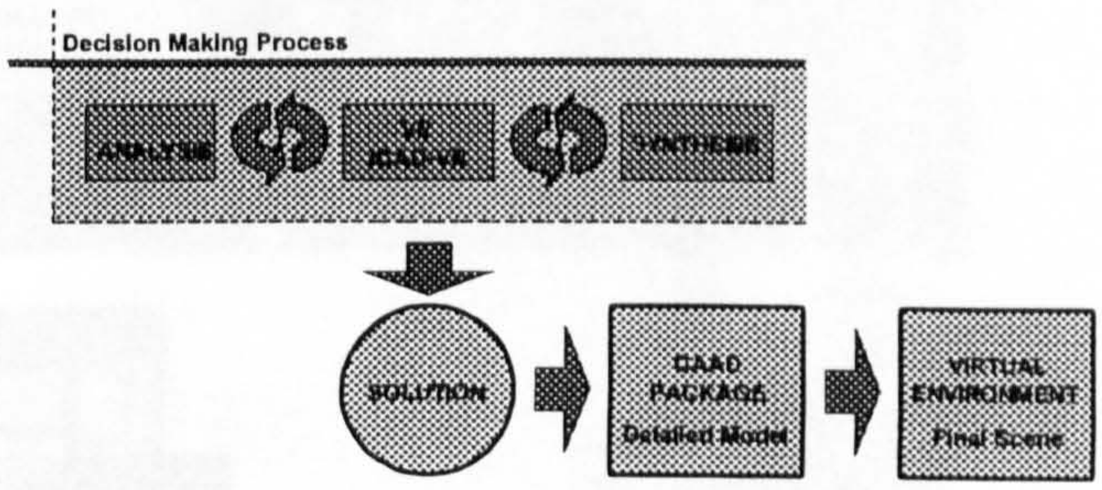


Figure 2. Proposed scenario

The system named JCAD-VR provides a flexible user-friendly immersive environment to support collaborative design on a synchronous base. It can be thought of as an investigation tool that allows the designer to sketch freely 3D objects within the virtual context. Moreover design solutions are shared in a synchronous fashion with other participants through the system's network-based architecture. Figure 2 shows the proposed scenario using JCAD-VR within the decision making process.

Here JCAD-VR provides the means for a more effective use of VR bridging between the phases of *analysis* and *synthesis*. VR is now employed at the very beginning of the decision making process when it is most likely to help in finding better design solutions. Once a desired solution is achieved the task of the creation of a very detailed 3D model and the final VR scene is given to appropriate CAAD packages. Moreover the participants are able to investigate design solution through concurrent design and synchronous collaboration.

JCAD-VR: The System Architecture

The two ideas upon which JCAD-VR is being built are:

?? that all the users present in the virtual world have to be able to share the same virtual environment in a "transparent fashion"

?? user interface (UI), instead of the traditional menu/windows based layout, is part of the virtual world itself. Any element of the interface becomes an object belonging to the 3D world and therefore it is treated as any other object. Each element of the interface can then be moved or scaled according to the user's needs.

The entire project is based on client-server architecture where every user logs into a virtual world and starts sharing design tasks with other users.

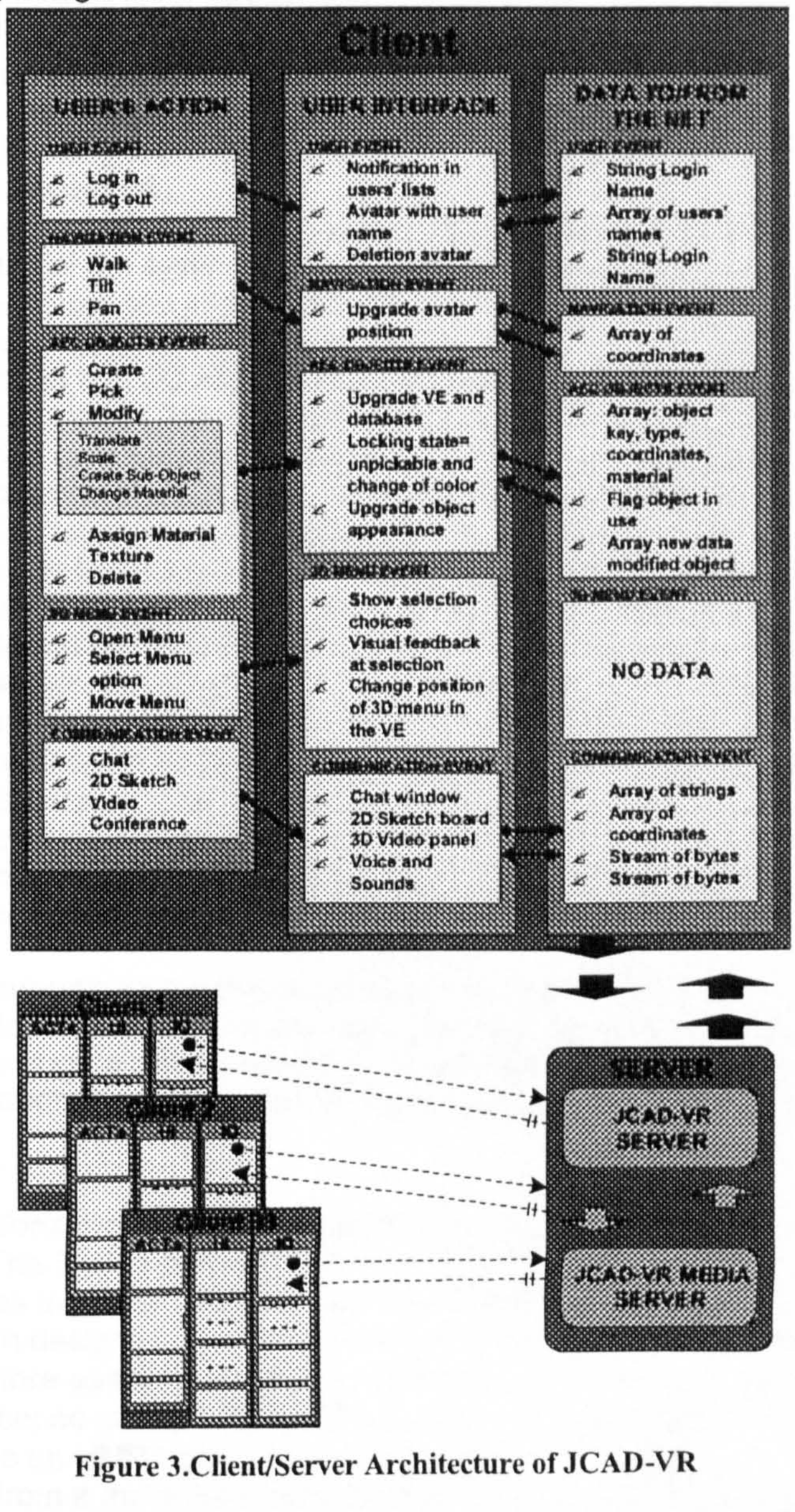


Figure 3.Client/Server Architecture of JCAD-VR

JCAD-VR is organised in an object-oriented fashion, where each module is able to fulfil certain task and it is independently coded. This approach has allowed the delivery of an initial functioning core of the JCAD-VR system, which will be expanded in the near future by adding several modules currently under development.

The system has been entirely developed around a client-server architecture to allow constant synchronous collaboration between several users. Every user accesses the virtual world, interacts with the VE and shares design tasks.

The Collaborative Approach

When JCAD-VR is initiated, the user is asked for a login name for the session and through an options panel he/she can decide which server to connect to and through which server port. This name will be used to identify participants and to communicate within the virtual world.

The system can be initiated in single mode or multiple screen mode. Single mode is set for the display device which consists of a standard computer screen; the multiple screen option has been included to allow devices such as the multi-projector display system processing the visual output of a Reality Centre which was used for the experiments.

In this phase it is also possible to activate or de-activate video conferencing facilities for the session. In instances when video conferencing is activated, support for video capturing device recognition and checking is provided. JCAD-VR provides also a stand-alone option in case collaboration is not required.

Once the system is initialised every window disappears freeing the space for the 3D graphic user interface (GUI) of the system. A set of 3D menus and icons appear on the screen and through them each user can interact with the system and with the other participants. A number of functions can be accessed through these menus, such as navigation and creation of objects. A number of 3D shapes and 3D AEC objects can be created and shared with other participants. The objects created can be following: geometric primitives (cones, boxes, spheres etc.) and architectural entities (walls, doors, windows etc.). The system routinely checks for constrains and allows only the possible modifications; for example a door cannot be moved onto or too close to another door. A "3D ruler" and a 3D panel close to the object are constantly providing the user with feedback related to the parameters which can be edited such as size, materials and cost.

The architecture of the system has been developed to allow every object created in the system to be assigned with a unique id-number. The ID is a combination of local ID and a user ID assigned by the server. In this way each object is attributed a unique number consistent for all the users in the system. When any object is selected by the user, this object is *locked* and such event is sent through the network to other users. Every time the user is about to modify an object this is checked against a network lock mechanism. This mechanism controls that several participants are not editing the same object at the same time and is designed in order to ensure consistency throughout the system. The system notifies every user internal database of any creation or modification of geometric objects within the virtual scene and broadcasts their numerical information.

To ensure communication between users, represented in the 3D world by avatars, different means are provided, from basic chat to voice and video conferencing. Freehand sketching in 2D is also possible through a shared electronic whiteboard. This architecture allows a real synchronous collaborative design making designing a true multi-user collaborative experience.

The Interface

Besides the functions provided by the system a great deal of attention has been put into the visual interface. The GUI or perhaps 3DGUI is in fact thought of as part of the virtual world itself.

This choice was made for two main reasons. From the technical point of view once the interface has been designed, it becomes independent from the visualisation device used. The system can therefore easily be adapted to different devices by just rewriting the code that is handling the device; no matter whether the system is running on a simple screen, on a Reality Center or linked to an HMD - the interface will always be in place.

Furthermore, from a more theoretical point of view, the interface becomes one of the elements of the virtual world and therefore it can be treated as any other object. Elements of the GUI such as panels, icons, rulers, are treated just like any other 3D entities within the VE. For instance, in the case of the video conferencing panel, the video coming from the other users is continuously rendered as a texture on a 3D panel.

All these elements can be replaced, dragged, re-scaled for the convenience of the user and perhaps they even provide a higher degree of feeling of immersion. The user interacts with the objects through elements of this interface, such as arrows placed to help the user editing the object. Feedback is provided through the visual modification of the object itself in the scene.

The 3D engine just renders all the possible changes of the VE: movements of avatars, video conferencing streams rendered on 3D panels and, most importantly, creation and modification of objects created within JCAD-VR.

Server Package

The server is made of two parts: a module which looks after the VE information to be broadcast, and another module which takes care of media streams and video conferencing tools. Both parts constitute the server system and they are closely linked to each other.

As an independent part of the framework the server has an autonomous and simpler interface that provides primarily information about the network status and transfer rate. A number of components are envisaged such as the communication status, the users on line, VR shared environments and the quality of the broadcast video for conferencing. Since the clients are communicating through independent processes, a future enhancement will allow the server to deal with several VR environments simultaneously.

The intrinsic multi-platform nature of JCAD-VR, inherited from the language used for coding, allows the server to transmit data to a broad range of platforms, from normal PCs to the supercomputer running a Reality Centre, and leaves the research team the freedom to test the software with several operating systems. The communication channel ensures the link between server and clients through a TCP/IP network.

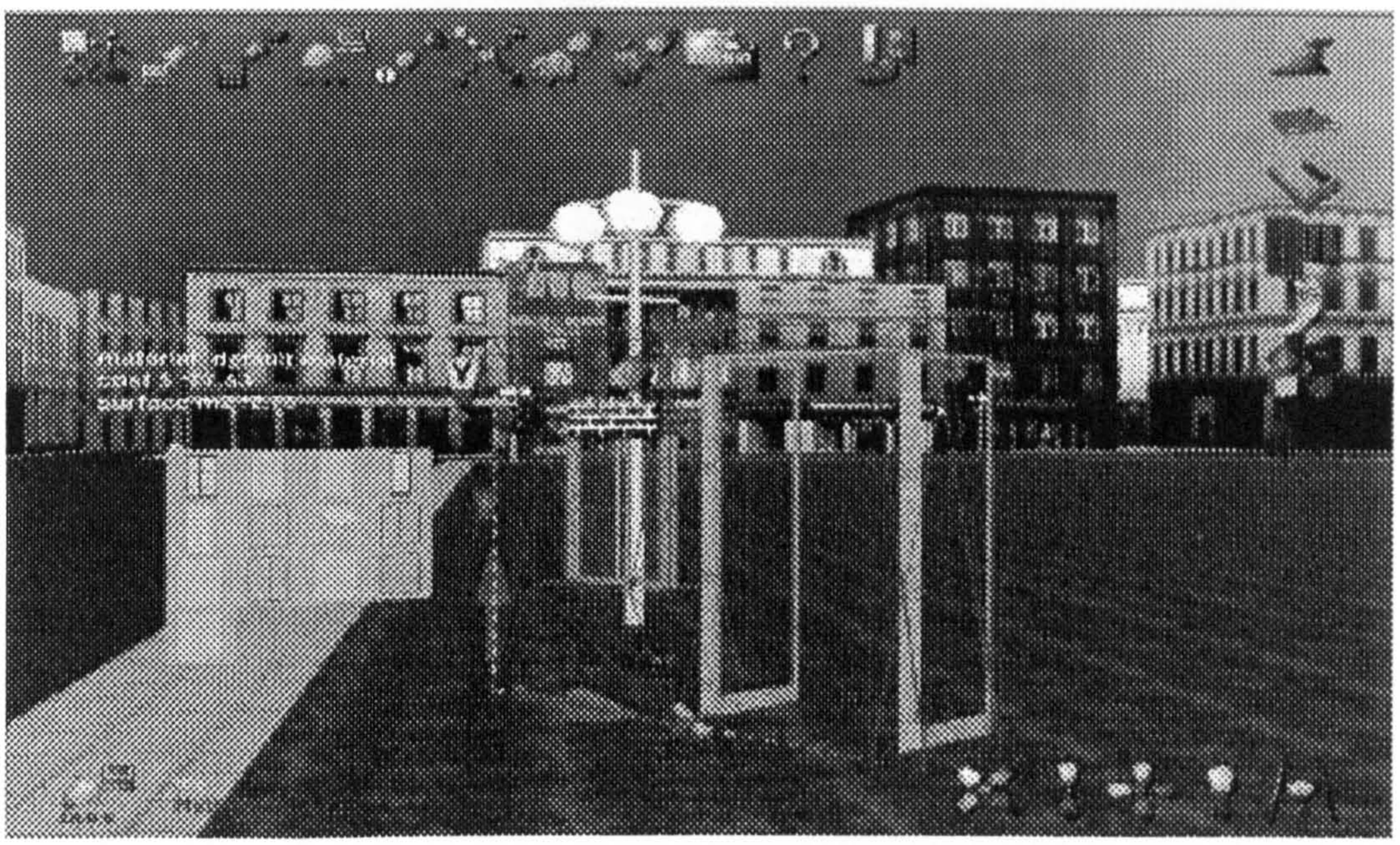


Figure 4. A screenshot from JCAD-VR

System Implementation and Hardware used

The whole framework of JCAD-VR was organised to allow concurrent software development, in a modular fashion, by individual members of the R+D team [3]. To facilitate this, an object oriented approach was identified as the most suitable one and the entire system was coded in Java? . The choice, even if less efficient in term of performances if compared with other choices, indeed offered great flexibility, true scalability and, last but not least, fully multi-platform support. Its network-centric nature, its multimedia integration together with the use of native graphic hardware and multi-processor support made it the obvious choice for the development of such real-time multimedia collaborative system.

The client application, in response to the obvious hardware limits imposed by the use of different hardware, has been written so that it can be easily customised to run on PCs as well as

on a Sgi supercomputers. The former are normal PCs whose video-card displays the virtual world only on a traditional window or at full screen. The latter is a 12-processors Sgi Onyx2 system running the Reality Centre at ABACUS, University of Strathclyde, Glasgow. When JCAD-VR is launched on the Sgi it can take advantage of its computational power to stretch itself on a 5 metre wide 2 metre tall tassellated screen where 3 Barco projectors create a 160 degree panoramic image.

The internal architecture of JCAD-VR is such that modules might be easily adapted to allow use of different VR devices such as CAVEs or Head-Mounted Displays, as well as several pointing devices such as a joystick, 3D mouse and VR Gloves. Further developments will include support for some of these devices.

From the collaborative point of view JCAD-VR is highly scalable and several communication media options are provided depending on the hardware limitations of the computer on which it is running.

The video conferencing facility has been coded using the Java™ Media Framework (JMF) which enables cross-platform capture, playback and streaming of audio and video at different transfer rates and resolutions. A great deal of effort has been expended by the research group to integrate the 3D module with the multimedia one.

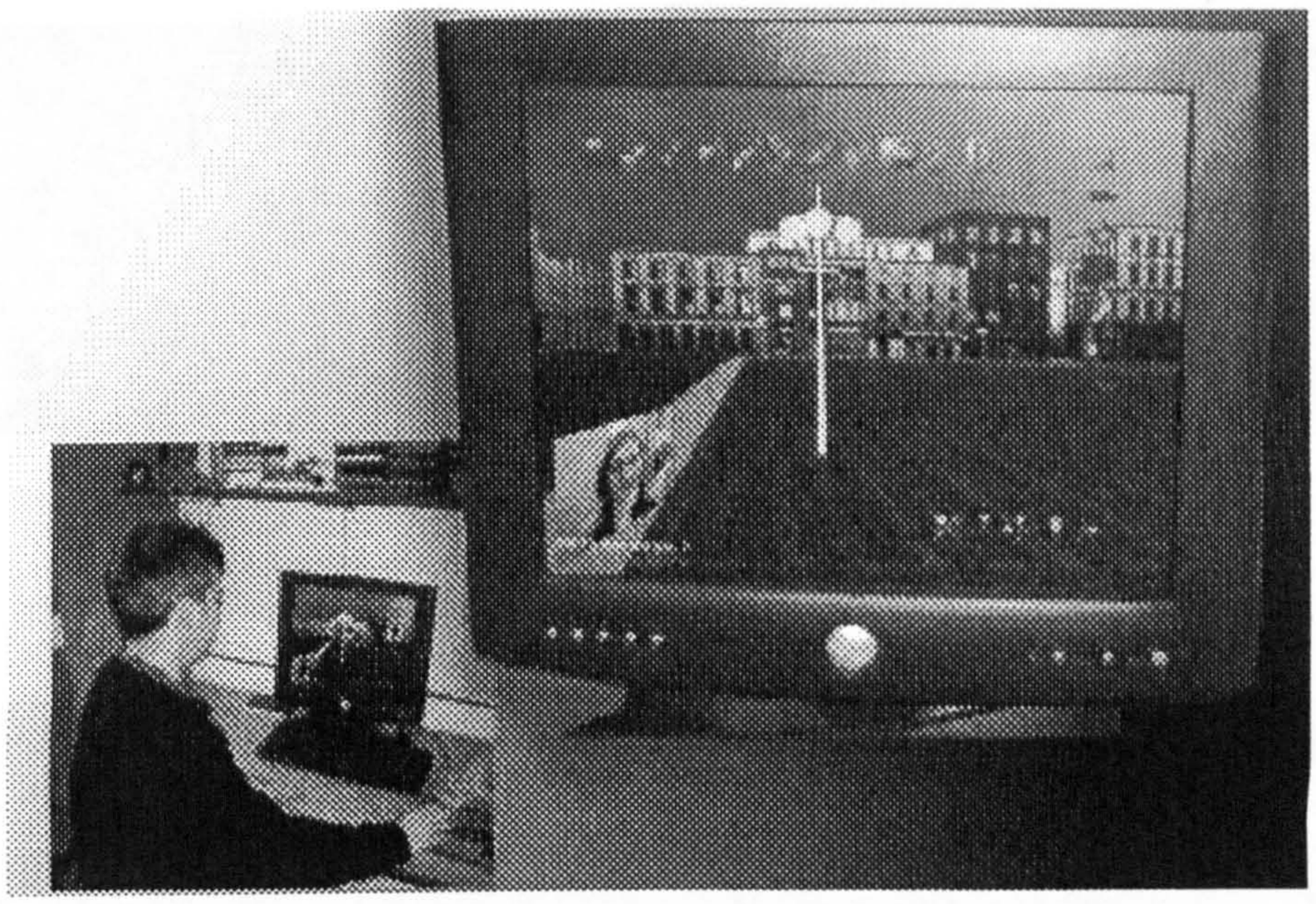


Figure 5. An image of JCAD-VR during an experiment of collaborative design

Collaborative Experiment

The obvious first line of inquiry regarding the usability and usefulness of the emerging system was in the academic environment within which it had been created. For the past three years, the academic with overall responsibility for CAAD teaching had offered an optional class, to fourth year students (and to students from less senior years with exceptional commitment and skills in CAAD) in the design application of innovative VR technologies. In Session 2001/2 three of thebstudents taking the class were introduced to JCAD-VR and invited to put the system to its first serious test.

Students Christoph Ackermann, Ross Marshall and Edward Wright were located, each with an appropriate workstation, in three different areas within the Department of Architecture, with fixed and hand-held video cameras covering the actions and observations of the students. Over the two-hour design session, the three students were invited to design an information centre in a public square and in a given urban context of Glasgow. The introduction to the project and to the specifics of the interface to JCAD-VR lasted a mere 30 minutes. This meagre introduction was purposeful and intended to test how intuitive (or not) the system was.

The in-house experiment was a revelation to the authors of this paper. Over the two-hour design period there was:

- ?? Fast and furious interaction amongst the three design participants within the common design environment; some 60/70 design scenarios were commonly generated, modified and agreed.
- ?? Both satisfaction and frustration amongst the participants was noted regarding the high degree of mutuality in the interactive process.
- ?? A real sense of having experienced a wholly immersive and shared design experience which heralds a future way of exploring and determining the configuration of the built environment.

The entire outcome of this limited experiment - with all its local and 'non-scientific' constraints - in common with the experiments in the late 1960' - is, thankfully, to stimulate further trials, tests and transformations.

The notion of a distributed design environment within which *all* can contribute, has, the authors claim, been significantly advanced.

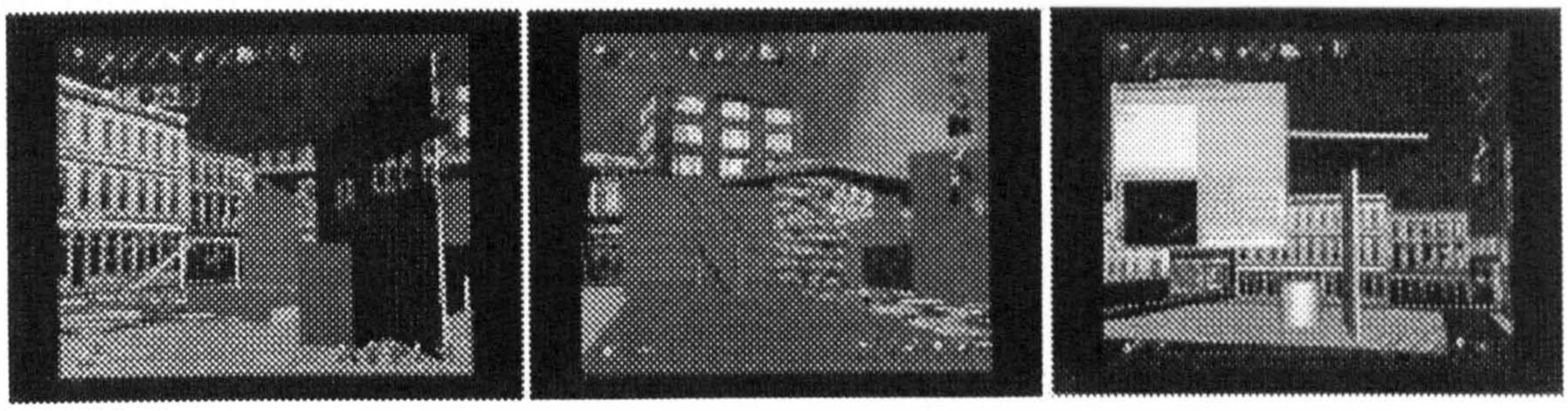


Figure 5. Screenshots from experiment

Figure 6. Pictures from the experiment

Conclusions and Further Developments

The prototype JCAD-VR system makes some steps toward the change of VR usage from mere presentation medium to a more powerful and effective design tool, and establishes the feasibility of VR becoming the interface for the next generation of computer aided design (CAAD) applications for architecture. Several enhancements are being considered for further development if the system including:

- ?? A voice driven interface enhancing friendliness of the user interface
- ?? Support for driving devices such as 6-degrees of freedom virtual glove

?? Implementation of a multi-environments server capable of dealing with several VR environments simultaneously.

References

[1] Barrilleaux, J. (2000) 3D User Interfaces with Java 3D, a guide to computer-human interaction in three dimensions. Manning, Greenwich.

[2] Bertol, D. (1997) Designing Digital Spaces, An architect's guide to Virtual Reality. John Wiley & Sons, Inc.

[3] Conti, G., Ucelli, G. and Maver, T. (2001) JCAD-VR - A Collaborative Design Tool in Virtual Reality, Proceedings of ECAADE 19. Helsinki, 2001, pp 454 – 459.

[4] Dorta, T. and LaLande, P. (1998) The impact of Virtual Reality on the design process, Proceedings from ACADIA Conference, Quebec City, 1998, pp 139 – 161

[5] Hughes, M., Hughes, C., Shoffner, M. and Winslow, M. (1997) Java Network Programming. Manning, Greenwich.

[6] Lawson, B. (1990) How designers think: the design process demystified. University Press, Cambridge.

[7] Sowizral, H. A. and Deering, M. F. (1999) The Java 3D API and Virtual Reality, IEEE Computer Graphics and Applications, May/June.

[8] Petric, J., Ucelli, G. and Conti, G. (2001) Educating the Virtual Architect, Proceedings of ECAADE 19. Helsinki, 2001, pp 388 – 393.

Conti, G., Ucelli, G. and Petric, J. (2002). JCAD-VR: a collaborative design tool for architects. In *Proceedings of the 4th international conference on Collaborative virtual environments, Bonn, Germany, 2002* (pp. 153-154). New York: ACM Press.

JCAD-VR: a collaborative design tool for architects

Giuseppe Conti
ABACUS
University of Strathclyde
Glasgow G4 0NG, Scotland, UK
Tel. +44 141 548 3407
giuseppe.conti@strath.ac.uk

Giuliana Ucelli
ABACUS
University of Strathclyde
Glasgow G4 0NG, Scotland, UK
Tel. +44 141 548 3407
giuliana.ucelli@strath.ac.uk

Jelena Petric
ABACUS
University of Strathclyde
Glasgow G4 0NG, Scotland, UK
Tel. +44 141 548 3407
j.petric@strath.ac.uk

ABSTRACT

Today the development of network-based virtual communities and the use of avatars have brought a new level of complexity to the meaning of virtuality, providing the technology for remote presence and collaborative experiences. In this project the intention was to pursue this articulated vision of Virtual Reality (VR) in order to assist all the participants - professional and client body - during the early stages of the design process. The objective was to provide a tool that is capable of creating 3D shapes in a shared VR environment, thus allowing the design and its evolution to be shared. The use of the Java™ programming language was a natural choice for this project. Because of Java™'s performance scalability and hardware independence the concept of CAAD has been extended, making it possible to create a VR environment that can co-exist between high-end supercomputers and standard PCs. The project is currently being tested using PCs workstations and an SGI system running a Reality Centre.

The research reported in this paper describes the development and application of software that aims to increase the opportunity for architects to collaborate within virtual worlds and enable effective and transparent information exchange.

Categories and Subject Descriptors

D.2.2 [**User Interface**]: 3D User Interface, H.5.3. [**Computer-supported cooperative work**] Collaborative Design, I.3.7 [**Virtual Reality**] Design system for Virtual Reality, J.6 [**Computer-aided design (CAD)**]

General Terms

Management, Design, Human Factors.

Keywords

Synchronous Collaborative Design, Virtual Environment, Interface, Architecture, Design Process.

1. INTRODUCTION

Historically architects have experienced the need to prove their design proposals using physical models. Due to the intrinsic spatial nature of the act of designing, even the most accurate 2D paper representation is usually not suitable to deliver the complexity of some architectonic ideas.

In the last decades the profession of the architect has been deeply affected by the *digital revolution* and the use of Computer Aided Architectural Design (CAAD) tools is nowadays part of the daily practice in most architecture firms. But in the last few years the "CAAD community" is experiencing a new revolution that is leading the move from static representation, based on 2D renderings or pre-recorded animations (considered as a sequence of 2D images), to dynamically generated 3D representations. Real-time navigation and interaction, typical of VR environments, provide just that fluent interface enabling the exploration of the design proposals that architects have not been able to get with any other media.

The increasing growth of computational resources and hardware power is probably preparing the anticipated transition to desktop VR applications, making them truly feasible tools for everyday practice. Furthermore, the recent growth of network-based virtual communities has brought a new level of complexity to the notion of virtual spaces, turning the profession of architect into something that might now resembles the one of the *virtual architect*.

Although VR is nowadays a quite mature technology, it is seldom used in architecture throughout the design process, but more often it is merely used as a powerful presentation technique.

Design methodologists in the past agreed on the need for iterative cycles between several phases of the design process. From studies concerning designers' behaviour [6] many authors observed an indefinite number of return loops from the moment when gathering of information and structuring of the design problem take place (known as *analysis*) to the one when design solutions are generated (known as *synthesis*).

The use of Virtual Reality within the design process could give to the designer an appropriate quick and practical response to his/her need of iteration and search for design solutions. Moreover it enables the capture of more information than would be possible to capture with the use of the traditional media and makes the checking of the design solutions more efficient by enhancing simulation capabilities. Furthermore VR broadens the boundaries of traditional perception by providing experiences of worlds not necessarily real or material. In short it is the perfect simulation medium for architects investigating design solutions.

It is then highly predictable that in the near future VR will become the interface for the next generation of computer aided drawing (CAD) applications and we can anticipate the change of its use from a mere presentation medium to a more powerful and effective design tool.

CAD/CAAD packages are very powerful but often complex rendering tools, which were not meant to be investigation tools, and therefore generating 3D models is often impractical and time-consuming. Therefore such models are usually employed when every design decision has been already taken.

Virtual environments (VE) are often created using CAAD packages for refinements, adjustments and exportation based on traditional 3D scenes. Documenting the evolution and development of design by constant updating of 3D models is an expensive business, and obviously even more expensive is the upgrading of the VEs generated from these models.

In these circumstances the use of VR would just increase costs. As result VR is relegated to the end of the design process rather then being used to engage design creativity through immersive design.

In the traditional scenario the decision making process does not take advantage of the technology but relegates the use of VR to the end of the process as a more convincing tool to impress contractors and clients. Only once the final solution has been achieved it is worth investing time into more powerful visualisation media.

Being aware of this background the research group engaged itself in the development of a VR system, named JCAD-VR, to help designers in the initial stages of the design process to take advantage of the VR as a new design tool.

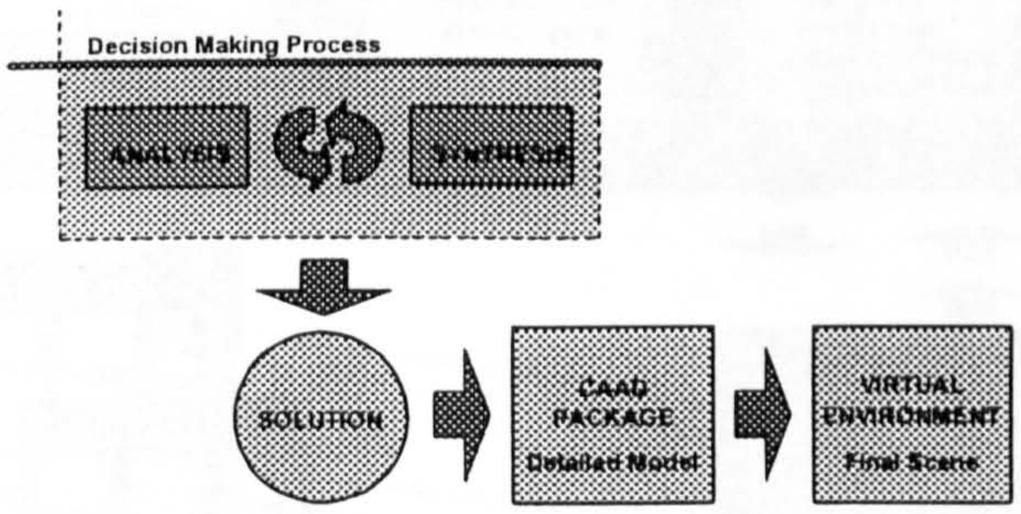


Figure 1. The current position of VR within the design process.

JCAD-VR provides a flexible user-friendly immersive environment to support collaborative design on a synchronous base.

It can be thought of as an investigation tool that allows the designer to sketch freely 3D shapes within the virtual context. Moreover design solutions are shared in a synchronous fashion with other participants through the system's network-based architecture. Figure 2 shows the proposed scenario using JCAD-VR within the decision making process.

Here JCAD-VR provides the means for a more effective use of VR bridging between the phases of *analysis* and *synthesis*. VR is now employed at the very beginning of the decision making process when it is most likely to help in finding better design solutions. Once a desired solution is achieved the task of the creation of a very detailed 3D model and the final VR scene is given to appropriate CAAD packages.

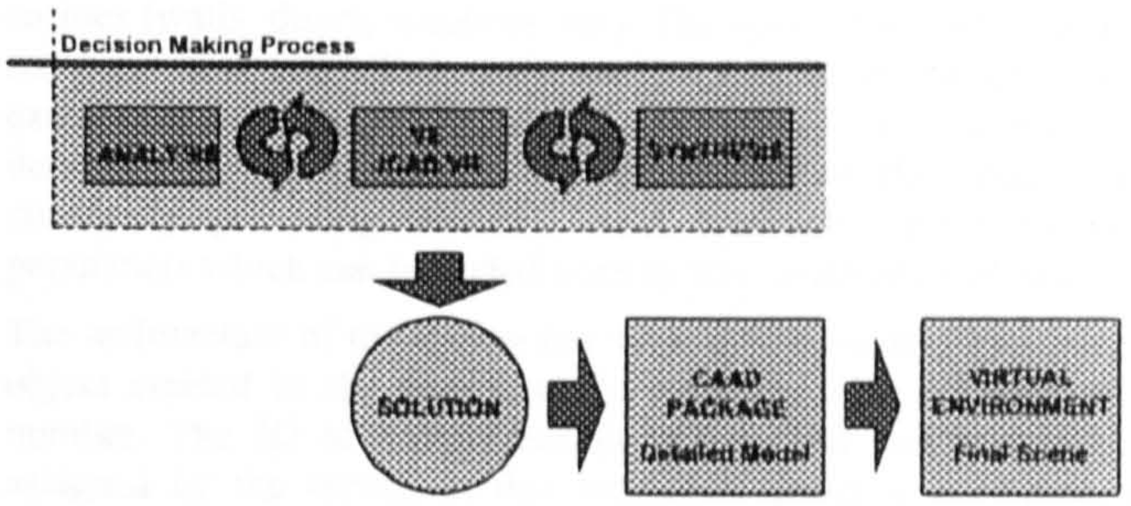


Figure 2. Proposed scenario

Moreover the participants are able to investigate design solutions through concurrent design and synchronous collaboration.

This paper reports the present state of the JCAD-VR system (Java™ based Collaborative Architectural Design tool in VR) and will highlight its future development and testing.

2. THE SYSTEM

The two ideas upon which JCAD-VR is being built are:

- that all the users present in the virtual world have to be able to share the same virtual environment in a "transparent fashion"
- user interface (UI), instead of the traditional menu/windows based layout, is part of the virtual world itself. Any element of the interface becomes an object belonging to the 3D world and therefore it is treated as any other object. Each element of the interface can then be moved or scaled according to the user's needs.

The entire project is based on client-server architecture where every user logs into a virtual world and starts sharing design tasks with other users.

JCAD-VR is organised in an object-oriented fashion, where each module is able to fulfil certain task and it is independently coded. This approach has allowed the delivery of an initial functioning core of the JCAD-VR system which will be expanded in the near future by adding several modules currently under development.

3. SYSTEM ARCHITECTURE

The system has been entirely developed around a client-server architecture to allow constant synchronous collaboration between several users. Every user accesses the virtual world, interacts with the VE and shares design tasks.

3.1 Client Side

3.1.1 The Collaborative approach

When JCAD-VR is initiated, the user is asked for a login name for the session and through an options panel he/she can decide which server to connect to and through which server port. This name will be used to communicate within the virtual world.

The system can be initiated in single mode or multiple screen mode. Single mode is set for the display device which consists of a standard computer screen; the multiple screen option has been included to allow devices such as the multi-projector display system processing the visual output of a Reality Centre used for the experiments.

In this phase it is also possible to activate or de-activate video conferencing facilities for the session. In instances when video conferencing is activated, support for video capturing device recognition and checking is provided.

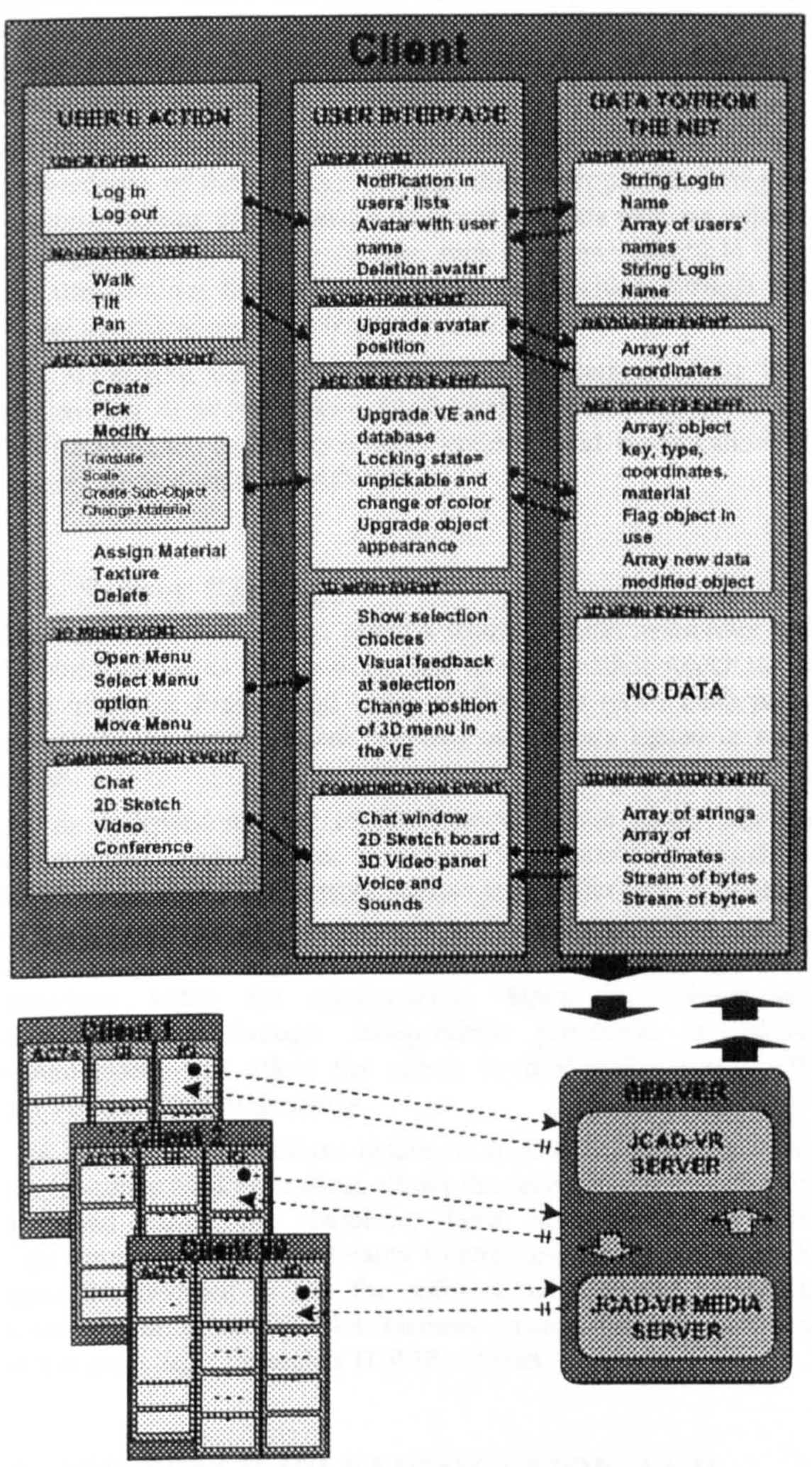


Figure 3.Client/Server Architecture of JCAD-VR

JCAD-VR provides also a stand-alone option in case collaboration is not required.

Once the system is initialised every window disappears freeing the space for the 3D graphic user interface (GUI) of the system. A set of 3D menus and icons appear on the screen and through them each user can interact with the system and with the other participants. A number of functions can be accessed through these menus, such as navigation and creation of objects. A number of 3D shapes and 3D AEC objects can be created and shared with other participants. The objects created can be the following: geometric primitives (cones, boxes, spheres etc.) and architectural entities (walls, doors, windows etc.). The system routinely checks for constrains and allows only the possible modifications - for example a door cannot be moved onto or too close to another door. A "3D ruler" and a 3D panel close to the object are constantly providing the user with feedback related to the parameters which can be edited such as size, materials and cost.

The architecture of the system has been developed to allow every object created in the system to be assigned with a unique id-number. The ID is a combination of local ID and a user ID assigned by the server. In this way each object is attributed a unique number consistent for all the users in the system. When any object is selected by the user, this object is *locked* and such event is sent through the network to other users. Every time the user is about to modify an object this is checked against a network lock mechanism. This mechanism controls that several participants are not editing the same object at the same time and is designed in order to ensure consistency throughout the system.

Figure 4. A screenshot from JCAD-VR

The system notifies every user internal database of any creation or modification of geometric objects within the virtual scene and broadcasts their numerical information.

To ensure communication between users, represented in the 3D world by avatars, different means are provided, from basic chat to voice and video conferencing. Freehand sketching in 2D is also possible through a shared electronic whiteboard. This architecture allows a real synchronous collaborative design making designing a true multi-user collaborative experience.

3.1.2 The interface

Besides the functions provided by the system a great deal of attention has been put into the visual interface. The GUI or perhaps 3DGUI is in fact thought of as part of the virtual world itself.

This choice was made for two main reasons. From the technical point of view once the interface has been designed, it becomes independent from the visualisation device used. The system can therefore easily be adapted to different devices by just rewriting the code that is handling the device; no matter whether the system is running on a simple screen, on a Reality Center or linked to an HMD - the interface will always be in place.

Furthermore, from a more theoretical point of view, the interface becomes one of the elements of the virtual world and therefore it can be treated as any other object. Elements of the GUI such as panels, icons, rulers, are treated just like any other 3D entities within the VE. For instance, in the case of the video conferencing panel, the video coming from the other users is continuously rendered as a texture on a 3D panel.

All these elements can be replaced, dragged, re-scaled for the convenience of the user and perhaps they even provide a higher degree of feeling of immersion. The user interacts with the objects through elements of this interface, such as arrows placed to help the user editing the object. Feedback is provided through the visual modification of the object itself in the scene.

The 3D engine just renders all the possible changes of the VE: movements of avatars, video conferencing streams rendered on 3D panels and, most importantly, creation and modification of objects created within JCAD-VR.

3.2 Server Package

The server is made of two parts: a module which looks after the VE information to be broadcast, and another module which takes care of media streams and video conferencing tools. Both parts constitute the server system and they are closely linked to each other.

As an independent part of the framework the server has an autonomous and simpler interface that provides primarily information about the network status and transfer rate. A number of components are envisaged such as the communication status, the users on line, VR shared environments and the quality of the broadcast video for conferencing. Since the clients are communicating through independent processes, a future enhancement will allow the server to deal with several VR environments simultaneously.

The intrinsic multiplatform nature of JCAD-VR, inherited from the language used for coding, allows the server to transmit data to a broad range of platforms, from normal PCs to the supercomputer running a Reality Centre, and leaves the research team the freedom to test the software with several operating systems. The communication channel ensures the link between server and clients through a TCP/IP network.

4. SYSTEM IMPLEMENTATION AND HARDWARE USED

The whole framework of JCAD-VR was organised to allow concurrent software development, in a modular fashion, by individual members of the R+D team [3]. To facilitate this, an object oriented approach was identified as the most suitable one and the entire system was coded in Java™.

The choice, even if less efficient in term of performances if compared with other choices, indeed offered great flexibility, true scalability and, last but not least, fully multi-platform support. Its network-centric nature, its multimedia integration together with the use of native graphic hardware and multi-processor support made it the obvious choice for the development of such real-time multimedia collaborative system.

The client application, in response to the obvious hardware limits imposed by the use of different hardware, has been written so that it can be easily customised to run on PCs as well as on a Sgi supercomputers. The former are normal PCs whose video-card displays the virtual world only on a traditional window or at full screen. The latter is a 12-processors Sgi Onyx2 system running the Reality Centre at ABACUS, University of Strathclyde, Glasgow. When JCAD-VR is launched on the Sgi it can take advantage of its computational power to stretch itself on a 5 metre wide 2 metre high tassellated screen where 3 Barco projectors create a 160 degree panoramic image.

The internal architecture of JCAD-VR is such that modules might be easily adapted to allow use of different VR devices such as CAVEs or Head-Mounted Displays, as well as several pointing devices such as a joystick, 3D mouse and VR Gloves. Further developments will include support for some of these devices.

From the collaborative point of view JCAD-VR is highly scalable and several communication media options are provided depending on the hardware limitations of the computer on which it is running.

The video conferencing facility has been coded using the Java™ Media Framework (JMF) which enables cross-platform capture, playback and streaming of audio and video at different transfer rates and resolutions. A great deal of effort has been expended by the research group to integrate the 3D module with the multimedia one.

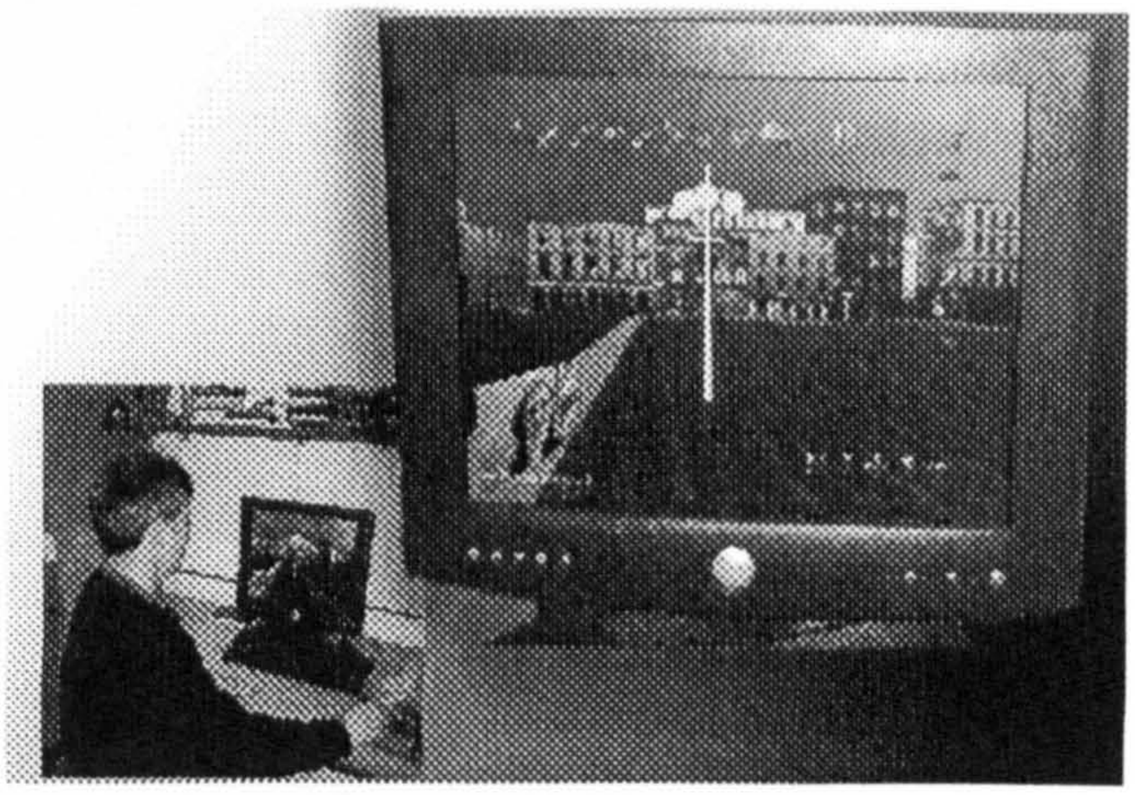


Figure 5. An image of JCAD-VR during an experiment of collaborative design

5. COLLABORATIVE EXPERIMENTS

A major test of the potential of JCAD-VR to contribute to collaborative design is currently underway. Under the auspices of ECAADE, an association of some 150 schools of Architecture within Europe, a bilateral experiment is taking place, between the Technical University of Eindhoven and University of Strathclyde

which seeks to establish, subjectively, if not statistically, the power of synchronous design support systems such as JCAD-VR.

The inter-institutional collaborative experiment will be informed by more local, simulated, distributed design decision exercises featuring committed students within the University of Strathclyde.

6. CONCLUSIONS AND FURTHER DEVELOPMENTS

The prototype JCAD-VR system makes some steps toward the change of VR usage from mere presentation medium to a more powerful and effective design tool, and establishes the feasibility of VR becoming the interface for the next generation of computer aided drawing (CAAD) applications for architecture. Several enhancement are being considered for further development of the system including:

- A voice driven interface enhancing friendliness of the user interface
- Support for driving devices such as 6-degree of freedom/virtual glove
- Implementation of a multi-environments server capable of dealing with several VR environments simultaneously.

7. REFERENCES

[1] Barrilleaux, J. (2000) 3D User Interfaces with Java 3D, a guide to computer-human interaction in three dimensions. Manning, Greenwich.

[2] Bertol, D. (1997) Designing Digital Spaces, An architect's guide to Virtual Reality. John Wiley & Sons, Inc.

[3] Conti, G., Ucelli, G. and Maver, T. (2001) JCAD-VR - A Collaborative Design Tool in Virtual Reality, Proceedings of ECAADE 19. Helsinki, 2001, pp 454 – 459.

[4] Dorta, T. and LaLande, P. (1998) The impact of Virtual Reality on the design process, Proceedings from ACADIA Conference, Quebec City, 1998, pp 139 – 161

[5] Hughes, M., Hughes, C., Shoffner, M. and Winslow, M. (1997) Java Network Programming. Manning, Greenwich.

[6] Lawson, B. (1990) How designers think: the design process demystified. University Press, Cambridge.

[7] Sowizral, H. A. and Deering, M. F. (1999) The Java 3D API and Virtual Reality, IEEE Computer Graphics and Applications, May/June.

[8] Petric, J., Ucelli, G. and Conti, G. (2001) Educating the Virtual Architect, Proceedings of ECAADE 19. Helsinki, 2001, pp 388 – 393.

Petric, J., Ucelli, G. and Conti, G. (2002). Participatory Design in Collaborative Virtual Environments. Accepted paper at the 6th SIGRADI Conference, Caracas.

Jelena Petric, Giuliana Ucelli, Giuseppe Conti
j.petric@strath.ac.uk , giuliana.ucelli@strath.ac.uk , giuseppe.conti@strath.ac.uk
ABACUS, Department of Architecture, University of Strathclyde, Glasgow, UK

Participatory Design in Collaborative Virtual Environments

Abstract

This paper re-establishes the theoretical framework for participatory design evolved in the late sixties and early seventies as part of the movement towards a more explicit design methodology and attempts an explanation of why the concept failed to gain commitment from the architectural and urban design professionals. The issue of user participation in the processes of building and urban design is enjoying renewed attention following its relative neglect over the last 20 years due, in large measure, to significant advances in emerging information technologies, particularly multimedia, virtual reality and internet technologies.

This paper then gives an account of two significant and relevant developments in the evolution of the application of information technologies with which the authors have been engaged. These are:

- a responsive and interactive interface to wholly immersive and realistic virtual reality representations of proposed buildings and urban neighbourhoods.
- an intuitive and platform-independent VR modelling environment allowing collaborative evolution of the scheme from within the virtual world.

The efficacy of these IT developments is tested in the context of a design exercise in which three designers, from distributed locations and using different computer platforms, collaboratively design an Information Centre from within the virtual world.

Resumo

Este trabalho restabelece uma estrutura teórica de projeto participativo originado no final dos anos 60 e início dos anos 70, como parte de um movimento direcionado a uma metodologia mais explícita de projeto e busca uma resposta para as razões do seu fracasso para a obtenção de um envolvimento maior dos profissionais das áreas de arquitetura e desenho urbano. A questão da participação do usuário nos processos projetuais arquitetônicos e urbanísticos está tendo uma renovada atenção após a relativa rejeição dos últimos 20 anos. Este fato deve-se, em grande parte, a avanços significativos em tecnologias da informação emergentes, particularmente multimídia, realidade virtual e tecnologias ligadas à Internet.

Assim, este trabalho pretende demonstrar dois desenvolvimentos significativos na aplicação da tecnologia da informação nos quais os autores estão envolvidos:

- uma interface sensível e interativa, completamente imersiva e representações realísticas de realidade virtual de edifícios e áreas urbanas.
- um ambiente de modelagem de Realidade Virtual, intuitivo e para múltiplas plataformas, permitindo a evolução colaborativa de um projeto a partir de um mundo virtual.

O impacto destes desenvolvimentos de TI é demonstrado a partir da realização de um projeto de uma estrutura de lazer para uma comunidade de usuários com deficiências físicas.

Design Decision Making

Architectural design is a multi-faceted occupation which requires, for its successful performance, a mixture of intuition, craft skills and detailed knowledge of a wide range of practical and theoretical matters. It is a cyclical process in which groups of people work towards a somewhat ill-defined goal in a series of successive approximations. There is no 'correct' method of designing and, although it is recognised that the process can be divided into separate phases, there is no generally accepted sequence of work that might guide design teams in the direction of achieving a satisfactory solution. Indeed, there are no solutions to design problems in the way that there are solutions to mathematical problems: the best that can be hoped for is an outcome which satisfies the maximum number of constraints which bound the area of concern. Furthermore, design is not an algorithmic process in which the desired

conclusion can be reached by the application of step-by-step procedures - first finalising this aspect, then that. It is a fluid, holistic process wherein at any stage all the major parts have to be manipulated at once. In this sense, it is less like solving a logical puzzle and more like riding a bicycle, blindfold, whilst juggling.

Despite the complexity of the design decision-making process the emerging new generation of computer-based models is already having an impact on how design is performed and, hence, on the quality of design. The impact stems from the fact that the new models, as opposed to paper-based plans and elevations or other conventional forms, are predictive rather than descriptive; dynamic rather than static; explicit rather than implicit and, above all, permit a more-or-less continuous and interactive assessment of a developing design on cost and performance.

Evidence is growing of the advantages offered by the application of computers in design, and these can be summarised as follows:

Widening the Search for Solutions

Access to programs which dynamically predict the cost and performances characteristics of optional design proposals can increase the scope of search for good solutions by as much as ten-fold. Not only is the search coverage extended, it is also more purposefully directed because designers are able to compare the quality of any one tentative solution against the quality of all previous solutions.

Greater Integration in Decision-Making

In conventional working, a great deal of design time is lost as proposals are passed to and fro between the architect (who tends to be the originator) and the other specialist members of the design team (who tend to the "checkers"). Quite frequently the scheme on which the architect has lavished time and effort is found by one or other of the specialists to be infeasible. With access to appropriate appraisal techniques embodied in computer programs, it is possible to check a proposal against a wide range of criteria from the outset of the design activity. Moreover, it is entirely practical (though not yet a widespread working method) for all members of the design team to have access to, and operate on, the common design model whether or not they share a design office. The models, then, can provide a strong integrating force in design team working.

Improving Design Insights

Apart from the use of appraisal programs to search for better designs, the programs can be used in a research and development context to provide insights into the way in which particular design decisions affect cost and performance. Typically, a designer working in this mode would select an existing building for study, then, keeping all other design variables constant (insofar as this is possible), systematically vary one factor while recording the cost/performance output from the program. In this manner, the architect can establish sets of causal relationships which provide powerful insights into structure of design decision-making.

Differentiation of Objective and Subjective Judgements

Contrary to the early fears of many architectural practitioners, the use of CAAD techniques focuses increased attention on subjective value judgements rather than less. As measurable attributes of optional designs are made more explicit, the necessary value judgements are forced to the surface of design activity and thereby, themselves become more explicit. The effect of this is to make it clear to designers and their clients, which judgements are based on quantifiable criteria and which on subjective and intuitive concepts.

Evidence of the degree to which computer-generated cost/performance information promotes effective value judgement, throws into sharp focus the crucial question: **whose** value judgement? This question was, for the first time, seriously addressed in the Design Participation Conference in Manchester in 1971. At that time, however, the human-machine interface was too primitive for the concept of useful participation by the users of buildings to be achieved. The new technologies of VR and Multimedia give real prospects for participation.

Virtual Reality

The essence of Virtual Reality (VR) is that the user, instead of looking through the window of his/her computer screen at a virtual world, can in effect, step through the window and enter the virtual world itself. This enhances:

- **Immersion**: Users are completely surrounded by the environment.
- **Presence**: Being surrounded the participant has actually the sensation of being *in* the environment. The Virtual Environment becomes then a place on its own and its perception is similar to real environments.
- **Interactivity**: This is surely the most important *feature* provided by VR: the environment allows the participant to be involved and the result of the actions done by the participant is visualized in the VE.

- **Autonomy**: Participants are neither constrained in paths nor in views preset by others but have the freedom and autonomy to explore any single part of the environment.
- **Collaboration**: Multiple users are able to take part and to interact in the same VE.

The effectiveness of VR for the *presentation* of design proposals is well established but its potential in the *process* of design has yet to be realized. However the speed at which technology is evolving is making the application of VR within the design professions a feasible approach. AEC companies have already started to evaluate how time consuming the traditional presentation path can be where animations or walkthroughs are used to show designs solutions to their clients. In fact traditional CAD/CAAD systems are used as rendering tools more than design tools. Any change on design solutions is subject to the inevitable delay of having to step back to the CAD/CAAD systems and then the result must be rendered again to be eventually visualized. This approach is obviously not only inconvenient but time consuming and therefore costly. The consequence of these issues is that some design and manufacturing companies have already started to investigate how VR can be used within the design process. The research and development reported in this paper hopefully makes a signal contribution to this investigation.

The JCAD-VR Concept

In the Department of Architecture and Building Science at the University of Strathclyde, the ABACUS group has been building a prototype design decision support system known as JCAD-VR.

The system can be initiated in single mode or multiple screen mode. Single mode is set for the display device which consists of a standard computer screen; the multiple screen option has been included to allow devices such as the multi-projector display system processing the visual output of a Reality Centre which was used for the experiments.

In this phase it is also possible to activate or de-activate video conferencing facilities for the session. In instances when video conferencing is activated, support for video capturing device recognition and checking is provided. JCAD-VR provides also a stand-alone option in case collaboration is not required.

Once the system is initialised every window disappears freeing the space for the 3D graphic user interface (GUI) of the system. A set of 3D menus and icons appear on the screen and through them each user can interact with the system and with the other participants. A number of functions can be accessed through these menus, such as navigation and creation of objects. A number of 3D shapes and 3D AEC objects can be created and shared with other participants. The objects created can be following: geometric primitives (cones, boxes, spheres etc.) and architectural entities (walls, doors, windows etc.). The system routinely checks for constrains and allows only the possible modifications; for example a door cannot be moved onto or too close to another door. A "3D ruler" and a 3D panel close to the object are constantly providing the user with feedback related to the parameters which can be edited such as size, materials and cost.

The architecture of the system (Figure 1) has been developed to allow every object created in the system to be assigned with a unique id-number. The ID is a combination of local ID and a user ID assigned by the server. In this way each object is attributed a unique number consistent for all the users in the system. When any object is selected by the user, this object is *locked* and such event is sent through the network to other users. Every time the user is about to modify an object this is checked against a network lock mechanism. This mechanism controls that several participants are not editing the same object at the same time and is designed in order to ensure consistency throughout the system. The system notifies every user internal database of any creation or modification of geometric objects within the virtual scene and broadcasts their numerical information.

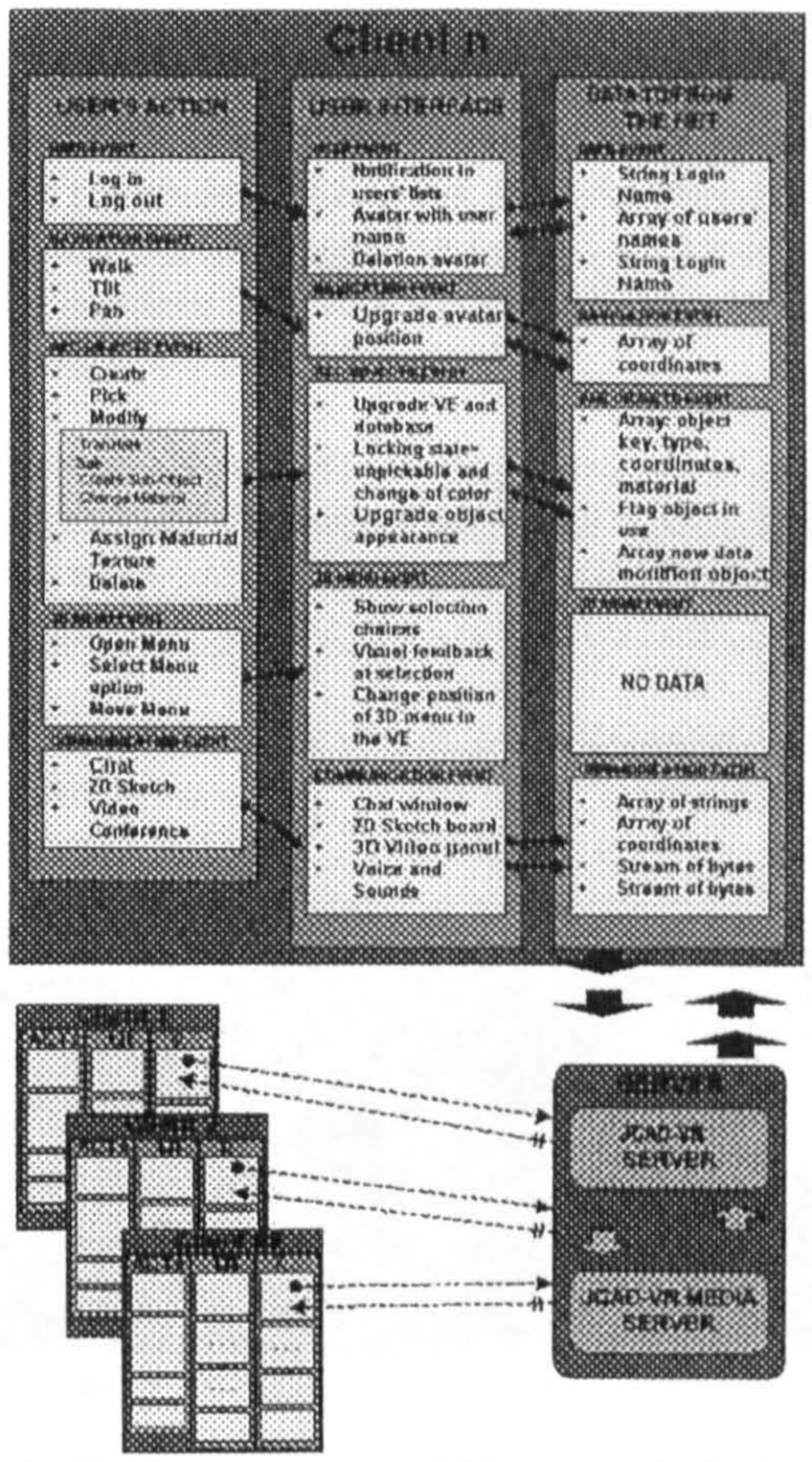


Figure 1 - Client/Server Architecture of JCAD-VR

To ensure communication between users, represented in the 3D world by avatars, different means are provided, from basic chat to voice and video conferencing. Freehand sketching in 2D is also possible through a shared electronic whiteboard. This architecture allows a real synchronous collaborative design making designing a true multi-user collaborative experience (Figure 2).

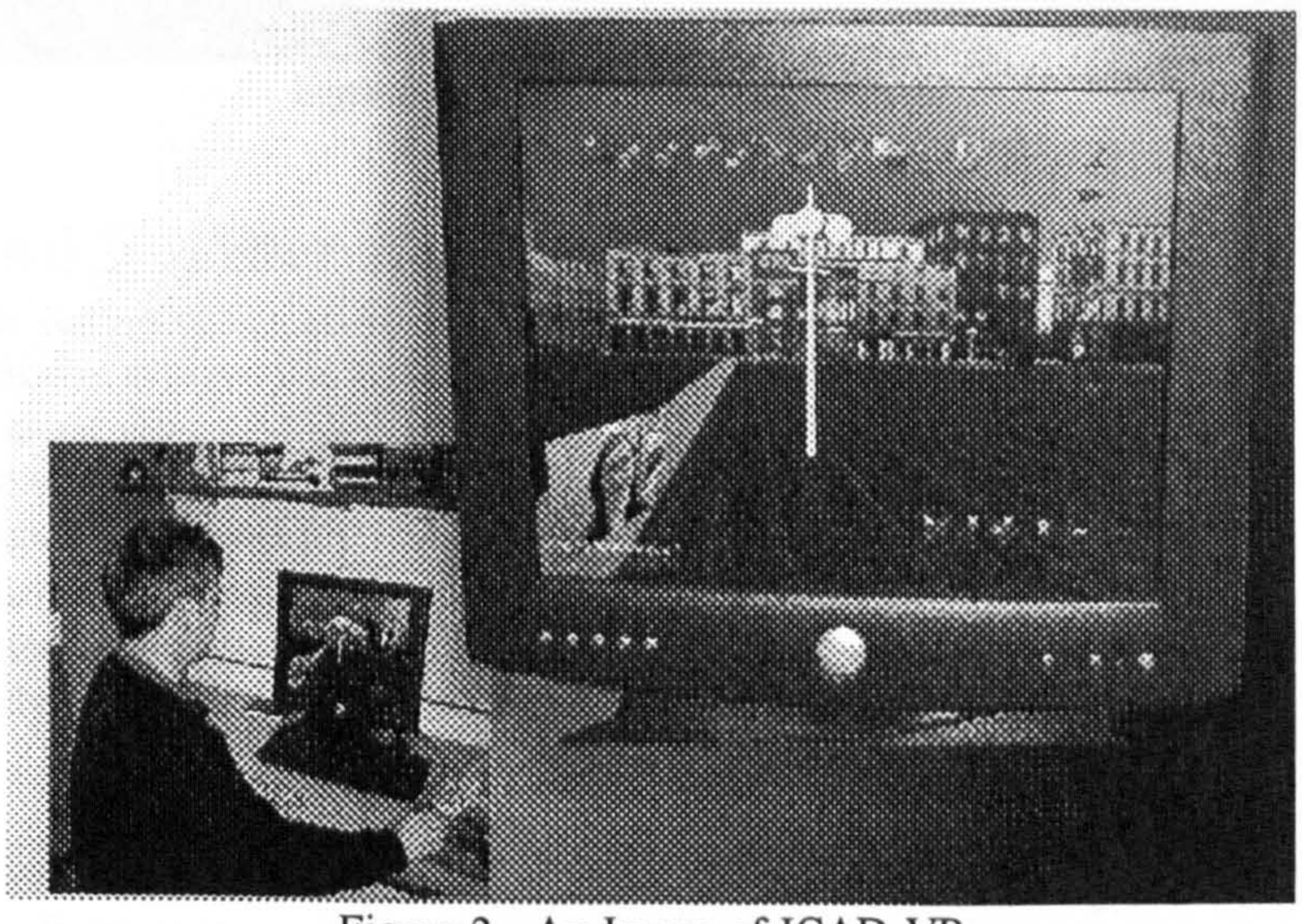


Figure 2 - An Image of JCAD-VR

Collaborative Experiment

The obvious first line of inquiry regarding the usability and usefulness of the emerging system was in the academic environment within which it had been created. For the past three years, the academic with overall responsibility for CAAD teaching had offered an optional class, to fourth year students (and to students from less senior years with exceptional commitment and skills in CAAD) in the design application of innovative VR technologies. In Session 2001/2 three of thebstudents taking the class were introduced to JCAD-VR and invited to put the system to its first serious test.

Students Christoph Ackermann, Ross Marshall and Edward Wright were located, each with an appropriate workstation, in three different areas within the Department of Architecture, with fixed and hand-held video cameras covering the actions and observations of the students. Over the two-hour design session, the three students were invited to design an information centre in a public square and in a given urban context of Glasgow. The introduction to the project and to the specifics of the interface to JCAD-VR lasted a mere 30 minutes. This meagre introduction was purposeful and intended to test how intuitive (or not) the system was.

The in-house experiment was a revelation to the authors of this paper. Over the two-hour design period there was:

- Fast and furious interaction amongst the three design participants within the common design environment; some 60/70 design scenarios were commonly generated, modified and agreed.
- Both satisfaction and frustration amongst the participants was noted regarding the high degree of mutuality in the interactive process.
- A real sense of having experienced a wholly immersive and shared design experience which heralds a future way of exploring and determining the configuration of the built environment.

Screen shots from the experiment are shown in Figure 3 and a frame from the video is shown in Figure 4.

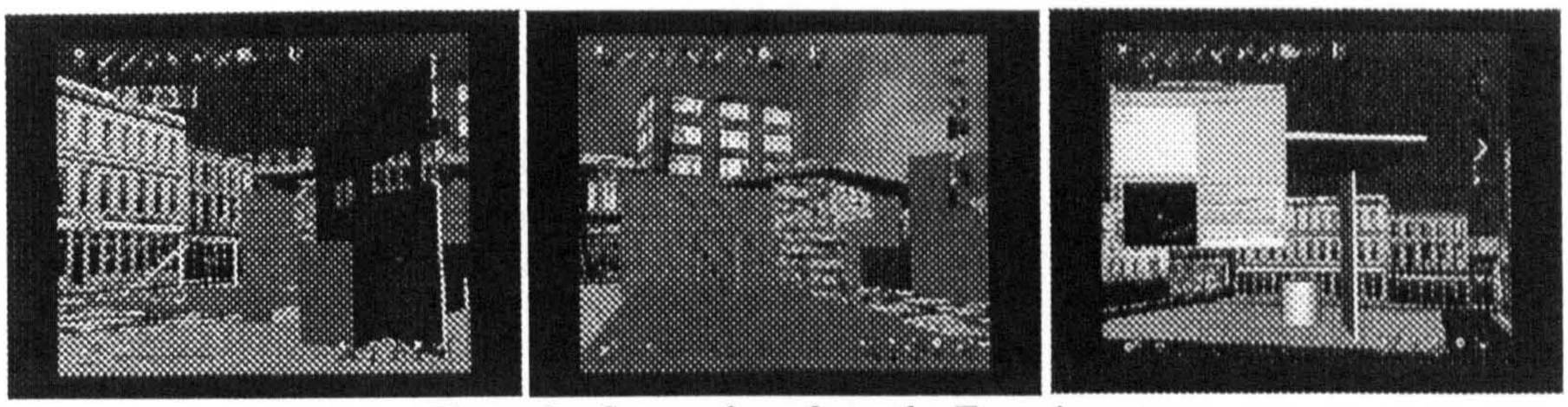


Figure 3 - Screenshots from the Experiment

Figure 4 - Pictures from the Experiment

Conclusions and Further Developments

The prototype JCAD-VR system makes some steps toward the change of VR usage from mere presentation medium to a more powerful and effective design tool, and establishes the feasibility of VR becoming the interface for the next generation of computer aided design (CAAD) applications for architecture. Several enhancements are being considered for further development if the system including:

- A voice driven interface enhancing friendliness of the user interface
- Support for driving devices such as 6-degrees of freedom virtual glove
- Implementation of a multi-environments server capable of dealing with multiple VR environments simultaneously.

References

Cross, N. (1971). *Design Participation*, Proceedings of the Design Research Society, Academy Editions.

Barrilleaux, J. (2000). *3D User Interfaces with Java 3D, a guide to computer-human interaction in three dimensions*, Manning, Greenwich.

Bertol, D. (1997). *Designing Digital Spaces, An architect's guide to Virtual Reality*, John Wiley & Sons, Inc.

Conti, G., Ucelli, G. and Maver, T. (2001). "JCAD-VR - A Collaborative Design Tool in Virtual Reality", Proceedings of ECAADE 2001, Helsinki, pp. 454 – 459.

Dorta, T. and LaLande, P. (1998). "The impact of Virtual Reality on the design process", Proceedings from ACADIA '98 Conference, Quebec City, pp. 139 – 161.

Hughes, M., Hughes, C., Shoffner, M. and Winslow, M. (1997). *Java Network Programming*, Manning, Greenwich.

Lawson, B. (1990). *How designers think: the design process demystified*, University Press, Cambridge.

Sowizral, H. A. and Deering, M. F. (1999), "The Java 3D API and Virtual Reality", IEEE Computer Graphics and Applications, May/June 1999.
Petric, J., Ucelli, G. and Conti, G. (2001), "Educating the Virtual Architect", Proceedings of ECAADE 2001, Helsinki pp. 388 – 393.

Petric, J. Conti, G. and Ucelli, G. (2003). Designing Within Virtual Worlds. Accepted paper at the CAAD Futures 2003 Conference, Tainan, Taiwan.

Designing Within Virtual Worlds

Jelena PETRIC, Giuseppe CONTI and Giuliana UCELLI
ABACUS, Department of Architecture, University of Strathclyde, Glasgow, UK

Keywords: Distributed Design, Virtual Environment, Collaboration, Interface.

Abstract: This paper celebrates the successful outcome of a trial of an innovative multi-platform distributed design decision support system in which the shared design environment exists within the virtual world. The outcome is the result of a sustained three-year research and development effort, within an internationally recognised research group. The project set itself a number of ambitious targets within the broad spectrum of distributed design decision support, viz:

- A multi-platform environment: the trial demonstrates inter-operability of different machine platforms - from a home PC to an international standard Virtual Reality Centre.
- A distributed environment: the trial demonstrates the high level of understanding amongst the design team separated by time and space.
- An ability to propose, discuss and agree upon, design decision from **within** the virtual world. Hitherto, virtual environments were viewing galleries; designers had to leave them to effect design changes in a conventional CAD package. The trial described in the paper amply demonstrates the potential to design, collaboratively and, in distributed mode, from **within** the virtual world.

The two ideas upon which the system (known as JCAD-VR) is built are:

- that all the users present in the virtual world have to be able to share the same virtual environment in a "transparent fashion";
- the user interface, instead of the traditional menu/windows based layout, is part of the virtual world itself. Any element of the interface becomes an object belonging to the 3D world and therefore it is treated as any other object. Each element of the interface can then be moved or scaled according to the user's needs.

The entire project is based on client-server architecture where every user logs into a virtual world and starts sharing design tasks with other users.

The authors propose to present a video which demonstrates the positive outcome of the trials to date. More importantly, perhaps, the authors will put the achievements of the R+D into the context of past aspirations and developments in the subject area and, most importantly of all, suggest how these modest achievements will impact on the next decade of increasingly rapid R+D.

1 CONCEPT

Historically, architects have experienced the need to prove their design proposals using physical models. Due to the intrinsic spatial nature of the act of designing, even the most accurate 2D paper representation is usually not suitable to deliver the complexity of some architectonic ideas.

In the last decades the profession of the architect has been deeply affected by the *digital revolution* and the use of Computer Aided Architectural Design (CAAD) tools is nowadays part of the daily practice in most architecture firms. But in the last few years the "CAAD community" is experiencing a new revolution that is leading the move from static representation, based on 2D renderings or pre-recorded animations (considered as a sequence of 2D images), to dynamically generated 3D representations. Real-time navigation and interaction, typical of VR environments, provide just that fluent interface enabling the exploration of the design proposals that architects have not been able to get with any other media.

The increasing growth of computational resources and hardware power is probably preparing the anticipated transition to desktop VR applications, making them truly feasible tools for everyday practice. Furthermore, the recent growth of network-based virtual communities has brought a new level of complexity to the notion of virtual spaces, turning the profession of architect into something that might now resembles the one of the *virtual architect*.

Although VR is nowadays a quite mature technology, it is seldom used in architecture throughout the design process, but more often it is merely used as a powerful presentation technique.

Design methodologists in the past agreed on the need for iterative cycles between several phases of the design process. From studies concerning designers' behaviour many authors observed an indefinite number of return loops from the moment when gathering of information and structuring of the design problem take place (known as *analysis*) to the one when design solutions are generated (known as *synthesis*).

The use of Virtual Reality within the design process could give to the designer an appropriate quick and practical response to his/her need of iteration and search for design solutions. Moreover it enables the capture of more information than would be possible to capture with the use of the traditional media and makes the checking of the design solutions more efficient by enhancing simulation capabilities. Furthermore VR broadens the boundaries of traditional perception by providing experiences of worlds not necessarily real or material. In short it is the perfect simulation medium for architects investigating design solutions.

It is then highly predictable that in the near future VR will become the interface for the next generation of computer aided drawing (CAD) applications and we can anticipate the change of its use from a mere presentation medium to a more powerful and effective design tool.

CAD/CAAD packages are very powerful but often complex rendering tools, which were not meant to be investigation tools, and therefore generating 3D models is

often impractical and time-consuming. Therefore such models are usually employed when every design decision has been already taken.

Virtual environments (VE) are often created using CAAD packages for refinements, adjustments and exportation based on traditional 3D scenes. Documenting the evolution and development of design by constant updating of 3D models is an expensive business, and obviously even more expensive is the upgrading of the VEs generated from these models.

In these circumstances the use of VR would just increase costs. As result VR is relegated to the end of the design process rather then being used to engage design creativity through immersive design.

In the traditional scenario the decision making process does not take advantage of the technology but relegates the use of VR to the end of the process as a more convincing tool to impress contractors and clients. Only once the final solution has been achieved it is worth investing time into more powerful visualisation media.

Being aware of this background the research group engaged itself in the development of a VR system, named JCAD-VR, to help designers in the initial stages of the design process to take advantage of the VR as a new design tool.

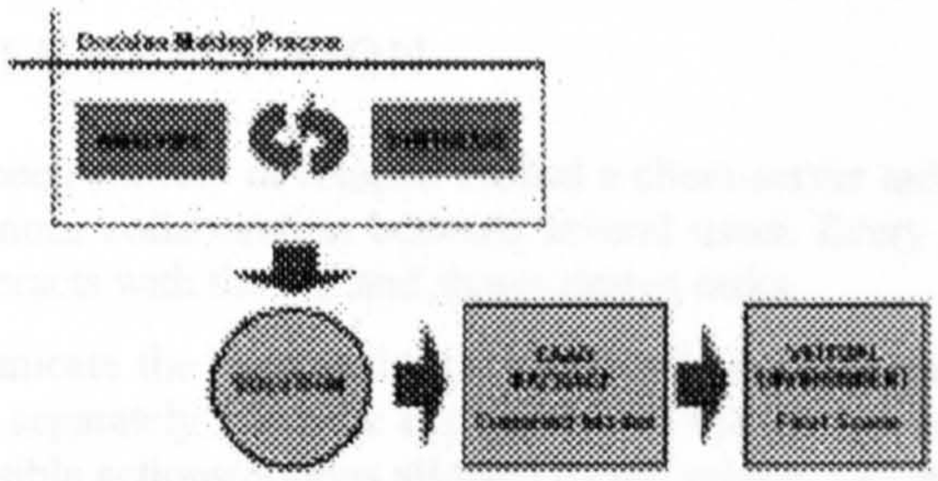


Figure 1 The current position of VR within the design process

JCAD-VR provides a flexible user-friendly immersive environment to support collaborative design on a synchronous base.

It can be thought of as an investigation tool that allows the designer to sketch freely 3D shapes within the virtual context. Moreover design solutions are shared in a synchronous fashion with other participants through the system's network-based architecture. Figure 2 shows the proposed scenario using JCAD-VR within the decision making process.

Here JCAD-VR provides the means for a more effective use of VR bridging between the phases of *analysis* and *synthesis*. VR is now employed at the very beginning of the decision making process when it is most likely to help in finding better design solutions. Once a desired solution is achieved the task of the creation of a very detailed 3D model and the final VR scene is given to appropriate CAAD packages.

Moreover the participants are able to investigate design solutions through concurrent

design and synchronous collaboration.

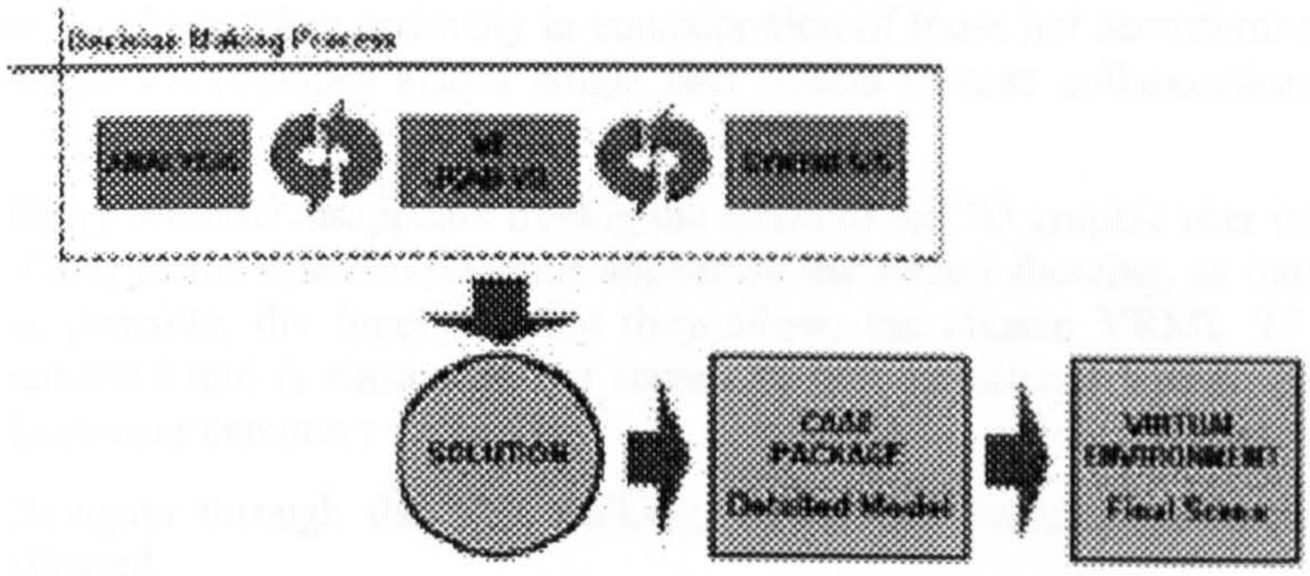


Figure 2 Proposed scenario

This paper reports the present state of the JCAD-VR system (Java™ based Collaborative Architectural Design tool in VR) and will highlight its future development and testing.

2 IMPLEMENTATION

The system has been entirely developed around a client-server architecture to allow constant synchronous collaboration between several users. Every user accesses the virtual world, interacts with the VE and shares design tasks.

To better communicate the capabilities of JCAD-VR, the explanation of the client side is presented separately from the explanation of the server. As if in a software demo, all the possible actions/options allowed by the system will be identified.

2.1 Client Package

1) On initiating JCAD-VR, the user is asked for a login name for the session and through an options panel he/she can:

- Decide which server to connect to and through which server port (in case the software is running on a local server).
- Decide if running in single or multiple screen mode. Single mode is set in case the display device is a normal screen; the multiple screen option has been included to allow devices such as the multi-projector display system processing the visual output of a Reality Centre.
- Load a VRML 97 file as a VR scene, if any.
- Activate or de-activate video conferencing facilities for the session (in case video conferencing facilities have been activated, support for video

capturing device recognition and checking are provided).

Providing the information required for this initial window is quite straight forward. Ease of use has been taken seriously in consideration of those not accustomed to the system. JCAD-VR provides also a single user option in case collaboration is not required.

2) Every window disappears freeing the space to the 3D graphic user interface (GUI) of the system. 3D icons appear on the screen showing, as intuitively as possible, the functions that they allow; the chosen VRML 97 file is rendered and is shown on the screen as in a traditional VRML browser. Each user can now:

- Navigate through the VE: walking, tilting and panning movements are allowed.
- Observe other participants' movements through their 3D avatars in the VE. Avatars are made distinguishable from each other by having, nearby, a 3D text of the users' login name.
- Listen to the other participants' voices through the loudspeakers.
- Watch, in the virtual scene, the 3D panels showing the video captured and sent by other participants. Every 3D panel shows the participants' login name to make them clearly distinguishable. These 3D panels are intrinsic parts of the GUI of JCAD-VR. Instead of being conventional windows they are 3D entities within the VE. These panels, as well as all the elements of the 3D GUI, are moveable for the convenience of the user.
- Check in a monitor the local user's captured video that is streamed out to the server.
- Check the list of users.
- Chat with other participants. This option is provided to assure a certain degree of communication between the participants in case the video conferencing facility is de-activated; this is available even when the video conferencing has been set as active.
- Freehand sketch in 2D on a shared electronic whiteboard, the possibility to set colours is provided in order to ease distinction between participants' contributions.
- Create 3D shapes and 3D AEC objects: both geometric primitives (cones, boxes, spheres etc.) and architectural entities (walls, doors, windows etc.) are available. A "3D ruler" is provided to help the user in constructing objects.

The choice of a 3D GUI was made in response to the possibility that modules supporting several display devices such as CAVEs and head-mounted displays would be included. Not having a traditional windows/menus user interface JCAD-VR can be used freely in every display situation minimising the effort to customise it for each device.

3) The 3D engine of the system renders all the possible changes of the VE: movements of avatars, video conferencing streams on the 3D panels and, most importantly, creation and modification of objects created within JCAD-VR. Each user, concerning 3D objects, can:

- Observe every object created in the VE both those created locally and by other participants. The system upgrades the VE with the objects created by all the participants in a synchronous fashion. An identification routine is provided in order to give each object a unique ID number to avoid interference in each users' local database of objects.
- Pick every *active* object in the virtual scene. *Active* objects here are all the objects created within JCAD-VR and not geometry imported with the VRML 97 file. Objects originally part of the VRML 97 file are considered to be *passive* and are not pickable. Once an object is selected:

  a) User priority on selected objects is set by a distributed locking mechanism. Locked objects will be no longer pickable for other participants until unlocked and their locked status becomes apparent through change into a red colour.

  b) 3D icons, 3D panels with general dimensions and x, y ,z arrows are set visible to help the user operate on the object.

  c) Translation, rotation and scaling in every direction are allowed. These modifications are operated on the object by simple dragging of the arrow representing the x, y or z axis. A 3D ruler and a 3D panel provide simple visual feedback to check the modified dimensions. Some routines to constrain modifications allowed on AEC objects are included, for example a door cannot be moved onto or too close to a second door and vice versa.

  d) Change of material is supported. In the first instance objects are created with a default grey colour but a library of textures is provided.

  e) Deletion of the object is allowed.

- Observe the visual feedback of the locking engine mechanism in case one of the participants has selected an object.
- Observe upgrading of the VE in case any modification on one or more objects has been carried out by any participant.
- Check for information on all *active* objects in the VE through a local database of objects. Objects are divided by type and general geometrical information is provided such as length, width, height, volume, radius, material etc. The system notifies to every users' internal database any creation or modification of geometric objects within the virtual scene and broadcasts their numerical information.

Figure 3 shows, that for every action performed by the user, the consequential visual

feedback and data as sent/received to/from the server through the network.

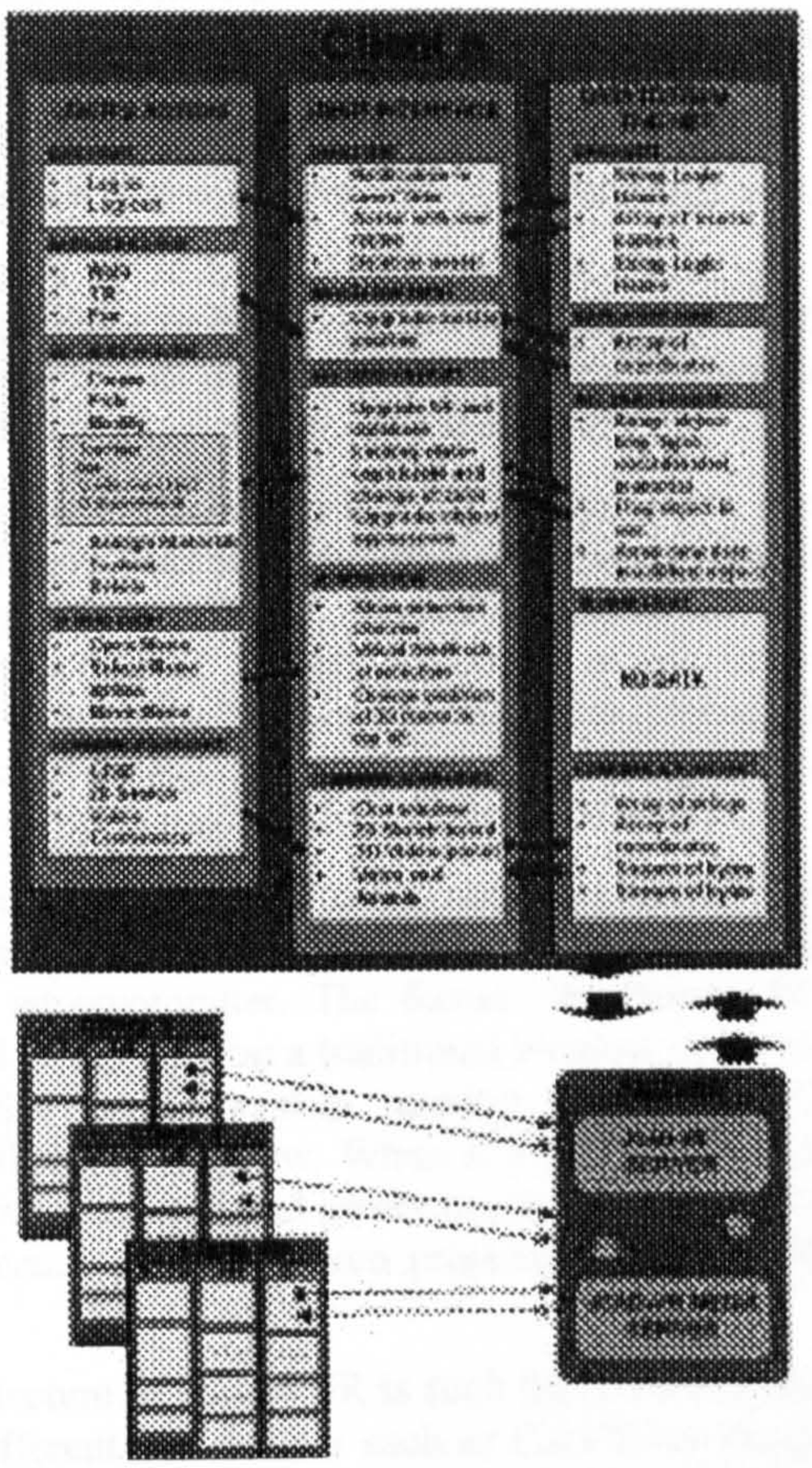


Figure 3 Client/Server Architecture of JCAD-VR

2.2 Server Package

The server is made of two parts: *JCAD-VR Server* which looks after the VE information to be broadcast, and *JCAD-VR Media server* which takes care of media streams and video conferencing tools. Both parts constitute the server system and they are closely linked to each other.

As an independent part of the framework the server has an autonomous and simpler interface that provides primarily information about the network status and transfer rate. A number of components are envisaged such as the communication status, the users on line, VR shared environments and settings and size of video conferencing windows. Since the clients are communicating through independent processes, a future enhancement will allow the server to deal with several VR environments

simultaneously.

The intrinsic multiplatform nature of JCAD-VR, inherited from the language used to code it, allows the server to transmit data to a broad range of platforms, from normal PCs to the supercomputer running a Reality Centre, and leaves the research team the freedom to test the software with several operating systems. The communication channel ensures the link between server and clients through a TCP/IP network.

The whole framework of JCAD-VR was organised to allow concurrent software development, in a modular fashion, by individual members of the R+D team (Conti et al. 2001). To facilitate this, an object oriented approach was identified as the most suitable one and the entire system was coded in Java™.

The choice, even if less efficient in term of performances if compared with some other languages, indeed offered great flexibility, true scalability and, last but not least, fully multi-platform support. Java3D™ was used to code the GUI and everything concerning the VE. Its network-centric nature, its multimedia integration together with the use of native hardware acceleration (OpenGL) and multi-processor support (in the case of Sgi workstation) make it the obvious choice for the development of a real-time multimedia collaborative system.

The client application, in response to the obvious hardware limits imposed by the use of different hardware, has been written to be easily customised to run on PCs as well as on a Sgi supercomputer. The former are normal PCs whose video-card displays the virtual world only on a traditional window or at full screen. The latter is a 12-processors Sgi Onyx2 system running the Reality Centre at ABACUS, University of Strathclyde, Glasgow. When JCAD-VR is launched on the Sgi it can take advantage of its computational power to stretch itself on a 5 metre wide 2 metre high tassellated screen where 3 Barco projectors create a 160 degree panoramic image.

The internal architecture of JCAD-VR is such that modules might be easily adapted to allow use of different VR devices such as CAVEs or Headmounted Displays as well as several pointing devices such as a joystick, 3D mouse and VR Gloves.

From the collaborative point of view JCAD-VR is highly scalable and several communication media options are provided depending on the hardware limitations of the computer on which it is running.

The video conferencing facility has been coded using the Java™ Media Framework (JMF) which enables cross-platform capture, playback and streaming of audio and video at different transfer rate and resolutions. A great deal of effort has been expended by the research group to integrate closely the two sections of JCAD-VR: the 3D module with the multimedia module.

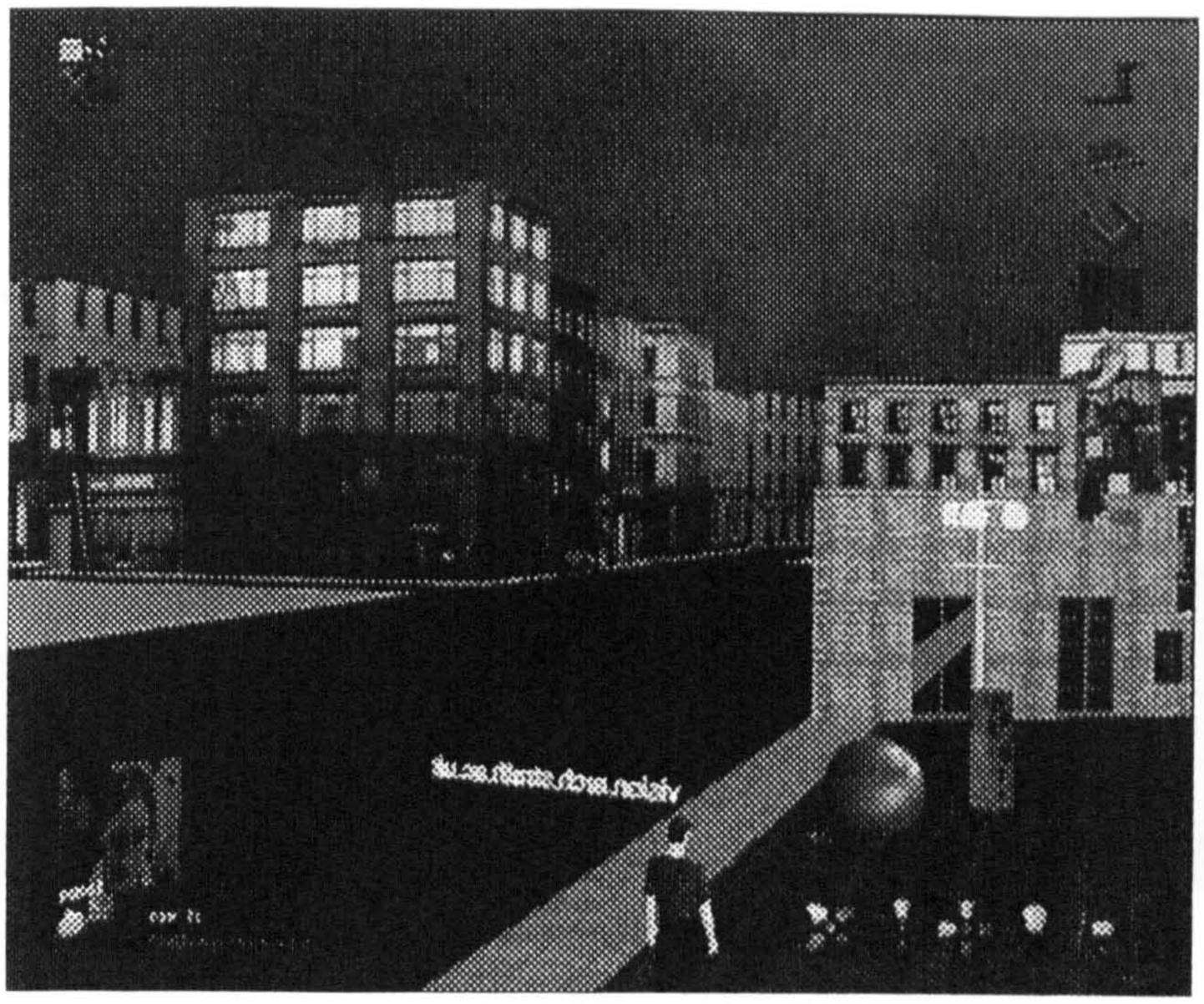


Figure 4 An Image of JCAD-VR

3 TRIALS OF THE SYSTEM

The obvious first line of inquiry regarding the usability and usefulness of the emerging system was in the academic environment within which it had been created. For the past three years, the academic with overall responsibility for CAAD teaching had offered an optional class to fourth year students (and to students from less senior years with exceptional commitment and skills in CAAD) in the design application of innovative VR technologies. In Session 2001/2 three of the students taking the class were introduced to JCAD-VR and invited to put the system to its first serious test.

Students Christoph Ackermann, Ross Marshall and Edward Wright were located, each with an appropriate workstation, in three different areas within the Department of Architecture, with fixed and hand-held video cameras covering the actions and observations of the students. Over the two-hour design session, the three students were invited to design an information centre in a public square and in a given urban context of Glasgow. The introduction to the project and to the specifics of the interface to JCAD-VR lasted a mere 30 minutes. This meagre introduction was purposeful and intended to test how intuitive (or not) the system was. The in-house experiment was rewarding to the authors of this paper. Over the two-hour design period there was:

- Fast and furious interaction amongst the three design participants within the common design environment; some 60/70 design scenarios were commonly generated, modified and agreed.
- Both satisfaction and frustration amongst the participants was noted regarding the high degree of mutuality in the interactive process.
- A real sense of having experienced a wholly immersive and shared design experience which heralds a future way of exploring and determining the configuration of the built environment.

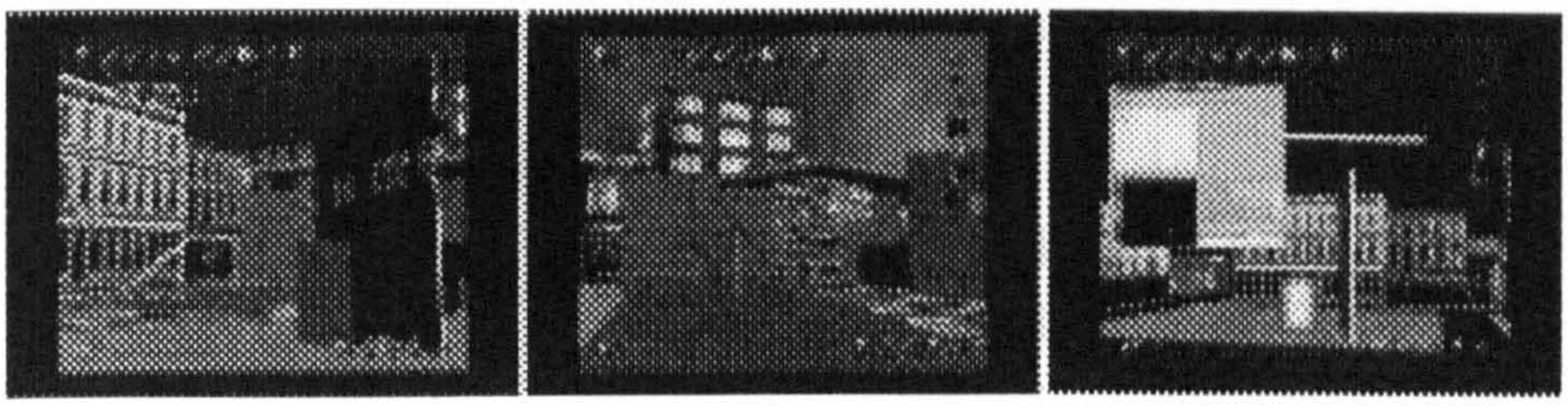


Figure 5 Screenshots from the Experiment

Figure 6 Pictures from the Experiment

4 CONCLUSIONS AND FUTURE DEVELOPMENTS

Assuming the availability of resources, it is intended to conduct further trials and development of the system.

Further trials will be trans-institutional and trans-national and the version of the system will include the recently implemented video-conferencing facility.

Further developments will include:

- A voice driven interface enhancing friendliness of the user interface;
- Support for driving devices such as 6-degrees of freedom virtual glove;

- Implementation of a multi-environments server capable of dealing with several VR environments simultaneously.

As in the earliest days of the introduction of computers into architectural design, the quantum jump is made by students. The work reported here, and which will be shown during the conference is, we believe, the epitome - in the current state of the art - of excellent practice. It makes a breakthrough, we believe, in the evolution of good design ideas and offers a real prospect for user participation.

There is some way to go, of course, to *design* interactively in a virtual environment. The next step which we envisage is to link to the 3D model the emerging and sophisticated software for the thermal, lighting and acoustics properties of the building. This would allow the user to visualise, dynamically, airflow, temperature gradients, lighting levels and to experience the actual acoustic characteristics of the space as she/he moves through it.

The other exciting development is for representatives of the client/user group to "join" the designer within the virtual environment and to participate directly in the evolution of the design concept.

BIBLIOGRAPHY

Barrilleaux, J. (2000). *3D User Interfaces with Java 3D, a guide to computer-human interaction in three dimensions,* Manning, Greenwich.

Bertol, D. (1997). *Designing Digital Spaces, An architect's guide to Virtual Reality,* John Wiley & Sons, Inc.

Conti, G., Ucelli, G. and Maver, T. (2001). JCAD-VR - A Collaborative Design Tool in Virtual Reality, *Proceedings of ECAADE 2001*, Helsinki, pp. 454 – 459.

Dorta, T. and LaLande, P. (1998). The impact of Virtual Reality on the design process, *Proceedings from ACADIA '98 Conference*, Quebec City, pp. 139 – 161.

Hughes, M., Hughes, C., Shoffner, M. and Winslow, M. (1997). *Java Network Programming*, Manning, Greenwich.

Lawson, B. (1990). *How designers think: the design process demystified*, University Press, Cambridge.

Petric, J., Ucelli, G. and Conti, G. (2001). Educating the Virtual Architect, *Proceedings of ECAADE 2001*, Helsinki pp. 388 – 393.