The Structured Development of Virtual Environments: Enhancing Functionality and Interactivity

Richard Eastgate

Thesis submitted to the University of Nottingham
for the degree of Doctor of Philosophy

September 2001

Abstract

Desktop Virtual Reality (VR) is an easy and affordable way to implement VR technology within an organisation. It provides an experience that can be shared by many people, and its 3D, interactive capability facilitates the communication of ideas not possible using other media formats. There are a number of software toolkits available for the building and programming of Virtual Environments (VEs), but very few resources that can help developers acquire the skills and techniques required to give their VEs utility and usability. This thesis reviews existing research into VE design with an emphasis on interactivity and usability, and then uses a case study based approach to conceptualise the VE development process and develop exemplar guidance tools.

The first group of case studies date from the early 1990s, with an emphasis on finding ways to build VEs incorporating functionality. The experience gained through these case studies was used to discover the issues most relevant to the VE developer and report on the techniques used to resolve them. Several models are then presented to explain these techniques and relate them to the VE development context. For the second set of case studies the emphasis moves to finding ways of making VEs more usable. Several approaches are presented and further conceptualisation results in a decision table based guidance tool.

The third set of case studies was carried out within the framework provided by the Virtual Environment Development Structure (VEDS), developed jointly by the author and other members of the Virtual Reality Applications Research Team (VIRART) at the University of Nottingham. In the light of this practical application of the framework and the experience gained throughout the case studies, changes are made to the structure to make it more accurately represent the actual process employed by VE developers. This version of VEDS is then used to more effectively define the areas where VE development guidance tools are needed. Using this information, and based on the experience acquired and the techniques developed throughout this research, three exemplar tools are presented.

Acknowledgements

Firstly, I would like to thank my supervisor, Professor John Wilson, for his guidance, help and encouragement throughout my PhD.

I would also like to thank all the collaborators on the case studies, Sue Cobb, Mirabelle D'Cruz, Sarah Nichols, Jo Crosier, Helen Haines, Victor Bayon, Helen Neale, and Rick Barnes, many of whom have also provided me with help and guidance throughout this research.

For all the help they've given me over the years I'd like to thank the people in the Virtual Reality Applications Research Team, the Institute for Occupational Ergonomics, and School4M at the University of Nottingham, especially Phillippa Scott, Harshada Patel, Amanda Ramsey, Steve Kerr, Garreth Griffiths, Alex Stedmon, Beverly Norris, Pam Soar, Lynne Mills and Anne Tigg. I would also like to thank the people at the Mixed Reality Lab.

Finally, I want to express my appreciation to my wife, Helen, and my children, Sam and Isaac, and the rest of my friends and family, for their patience, understanding and encouragement, especially during the writing up period.

Table of Contents

Glossary of Terms

The author has used these terms throughout the thesis and has given some of them specific meanings within the context of VE development. Many of them are explained in greater detail in the chapters but are also defined here for quick reference.

- **Affordances** – a property of an object (or other entity) that makes its function apparent
- **Client** – the person or body that commissions or requests the development of a VE
- **Desktop VR** – a VR system that uses a PC monitor as the display device.
- **Functionality** – the behaviours, dynamics and interactivity of virtual objects and systems. Whilst including direct user interactions such as object manipulation, functionality also includes indirect consequences of user interactions (e.g. virtual machines performing semi-automatic operations), and autonomous processes (e.g. a clock on the wall) within a VE.
- **GUI** – Graphical User Interface
- **HMD** – Head Mounted Display
- **Hybrid interface** – a VE user interface that incorporates 2D and 3D components. Often the 2D components are overlaid on or around a window onto the 3D environment.
- **Input device** – Piece of hardware used to interact with a VE, for example a joystick or a mouse.
- **Interaction** – reciprocal activity, any action by the user that results in changes to the VE that could be sensed by the user.
- **Manipulation** – using the VR user interface to move or change objects in a VE.
- **Navigation** – finding ones way, building up a mental map of an environment, using that map to plan a route to a location, and then moving the viewpoint to that location.
- **Object** – an entity in a VE, could be made up of many shapes.
- **Object behaviour** – broad term to describe the properties and functionality of an object that determine the way that the object acts.
- **Object dynamics** – broad term used to describe movement of an object in a VE.
- **Picon** – Picture ICON, used as part of a hybrid interface to provide feedback as to the status of objects or systems in a VE.
- **Shape** – an elementary component of a VE.
- **Topology** – the relative positions and structure of shapes and objects in a VE.
- **User** - a person experiencing a VE.
- **VE** – Virtual Environment, a computer generated 3D interactive model.
- **VE developer** - a person who creates the 3D models and programs the behaviours and dynamics of a VE.
- **VEDS** – Virtual Environments Development Structure
- **VET** – a Virtual Environment Training application.
- **Viewpoint** – a position, usually controlled by the user, from which a VE is viewed (see navigation). Sometimes known as a 'camera'.
- **VIRART** – the Virtual Reality Applications Research Team, based at the University of Nottingham
- **VR** - Virtual Reality, the technology that is used to create and present VEs
- **WIMP** – a 2D user interface incorporating Windows, Icons, Mice, and Pointers.

Chapter 1 Introduction

1.1 Background

As Virtual Reality (VR) technology becomes more widely used the question of 'whether and why we should use VR' decreases in importance and is replaced by 'where and how should we use VR?' The author was a founder member of VIRART (the Virtual Reality Applications Research Team) in 1991, and since then he has been developing virtual environments (VEs) for industry, education and research purposes. As a part of this process he has become familiar with many of the problems encountered during the development of VEs and the techniques that can be adopted to solve them. Much of the research undertaken at VIRART involves the identification and development of practical VE applications within the limitations of the current technology. These applications are implemented and evaluated in practical settings such as educational or industrial environments. This applications oriented approach is driven by close collaboration with client organisations, such as industrial companies (e.g. Rolls-Royce, Unilever and NCR), educational bodies and charities. The process used provides the opportunity for iterative development and evaluation of both individual applications and also of the overall approach to VE development, via contact with users, either in the workplace or during experimental programmes. In addition the extended practical programme has given great opportunities to carry out fundamental research into the nature of VEs and into understanding appropriate design.

This thesis documents a number of VE applications development case studies and goes on to describe the techniques and strategies derived from those case studies. The case studies mostly deal with the development of VE applications for education and training in which the author was not only the programmer but also played a significant role in the specification and design.

1.2 Defining the problem

Many of the issues confronting the VE developer stem from the fact that the VR system user interface is different to the traditional 2D computer interface in a number of significant ways. Using a traditional computer interface, the user interactions take place from outside the 2D environment. With VE technology the user can be cognitively, if not physically, immersed in the environment. This, and the 3D nature of the environment, lead to a number of factors that the VE developer has to take into account.

1. In a VE the range of possible types of interaction is much larger than those available on a traditional 2D interface (which are mainly clicking on buttons with the occasional drag of the mouse pointer).
2. In a VE points of interaction are likely to be representations of 3D objects. This leads to points 3, 4 and 5.
3. In the real world, these objects will have different interaction methods (buttons are pressed, handles are turned or pulled etc). Many real world interaction methods (such as applying your weight to open a swing door) will not be available in a VE because of the limitations of the VR system input devices. Therefore the implementation of interactions in a VE often requires the design of VE viable interaction methods to replace those used in the real world. This can result in inconsistent and/or non-intuitive interaction methods in a VE.
4. Interactions that appear to be available may not actually be available (some objects that are included purely to improve realism will be non-interactive).
5. From the user's viewpoint, points of interaction may be obscured or too distant or small to be seen easily. Without being guided towards these points of interaction the user may remain ignorant of their affordances.
6. Compared with navigation in a WIMP (Windows, icons, mice and pointers) environment, navigation in a VE can be non-linear, naturalistic, unstructured and inexact. For instance, in a WIMP environment, navigation has very limited degrees of freedom, usually only one; clicking on a button might bring up a specific dialogue box, whilst scrolling down usually takes you to the next part of a document. In a VE the user can navigate with up to six degrees of freedom, making the outcome of any movement much less predictable, much harder to structure, more approximate, but much more like real life.

7. In a VE, navigation prior to interaction requires movement to the point of interaction *and* then reorientation so that the point of interaction is visible. This is more difficult for the user than the equivalent activity in a WIMP environment, where the point of interaction may be a button inside a dialogue box. Navigation requires the opening of that dialogue box (most likely also done by clicking on a button). If the GUI (graphical user interface) has been designed properly the dialogue box will open on part of the screen that is visible to the user making it easy for the user to perform the interaction. On the other hand, in a VE as in the real world, one has to move close enough to an object to enable interaction. Then, in a VE but not necessarily in the real world, one has to be able to see the object in order to interact with it (for a specific example of this refer to section 5.3.2 of this thesis), i.e. the user must turn to face the point of interaction (it may be that with advanced haptic feedback devices seeing the object may not be necessary for interaction, but what is written here holds true for the vast majority of current VR/VE user interfaces).
8. Compared with 2D GUIs, VE interaction metaphors will be more complex, reflecting the more complex 3D nature of the user interface. For example, one of the criteria for the design of a WIMP environment is the use of a mouse and a keyboard as input devices. The outline geometry of a VE is often not designed at all but rather is a model of an actual, real world location, and as such may not lend itself to being easily interacted with in the real world, let alone the virtual.

The recognition of these and other differences between VR and traditional 2D user interfaces led Herndon to conclude that - *'Most have realised that 3D graphics applications are significantly more difficult to design than their 2D counterparts'* (Herndon, Van Dam, & Gleicher, 1994). The general lack of understanding of the demands of VE development has led to a situation where many VEs created are inappropriately designed (typically they are visually impressive but difficult to use). A successful VE package will use VR technology to best satisfy the needs of the organisation and the user. VIRART, with technical development led by the author, have adopted a multi-disciplinary approach incorporating not just an understanding of the technology, but also an appreciation of the domain in which the technology is to be applied and the characteristics and needs of the expected user population. This thesis aims to integrate across and to build on that research, by using the experiences gained from VE development case studies to provide structure and guidance for the VE development process.

1.3 The technology used

1.3.1 The Superscape Virtual Reality Toolkit (VRT)

Whilst the author has developed VEs using other VE development systems (e.g. V-Space by Virtuality, and DVise by Division), the case studies covered by this thesis were all developed using Superscape VRT. However, the experiences gained from these case studies are largely applicable to other desktop, and to a lesser extent, immersive VR technologies. The Superscape VRT is a VE development platform that runs on Microsoft operating systems (initially DOS up until 1995, and then Windows 95, 98 and NT). Whilst it can support immersive devices such as head mounted displays (HMDs) it particularly lends itself to the creation of VEs for desktop or flat screen VR. Its provision of simple collision detection and limited implementation of physical parameters such as friction, in combination with 3D rendering algorithms borrowed from the computer games world, make it a powerful tool for the rapid construction of functionally rich VEs to run on low power computing platforms. The toolkit has three main components; the shape editor (a 3D modeller), the world editor (where worlds are assembled and functionality is added), and a visualiser, which is used to experience the finished worlds. Other components are the image editor (used for creating and editing texture maps for 3D objects), the sound editor (for adding sound to the VE), the resource editor (for the creation of dialogue boxes), and the layout editor for the creation of hybrid user interfaces (a combination of VE and traditional 2D interfaces, best used on desktop VR systems). All of these VRT components use the same GUI format, making them easy to use.

1.3.2 VR system hardware

One of the advantages of desktop VR is that the hardware used can be a standard desktop PC system with a few optional extras. This results in a low cost way of implementing VR technology within an organisation. Also, over the last decade, the computing power of the standard 'off-the-shelf' desktop PC has increased faster than the processing demands of the average VE. Nowadays it is unusual to find a PC that is not powerful enough to render a well-designed Superscape VE. However, this was not always the case and in the early days of this author's work (using 486 PCs), maintaining an adequate frame rate was a major demand on the VE developer (this is documented later in this thesis).

Whilst a Superscape VE can be navigated and interacted with using the standard computer keyboard and mouse, this can be supplemented by the use of an inexpensive (about £20) joystick or the more expensive (about £500) spacemouse. Before the spacemouse was available the even more expensive (about £950) and less robust spaceball was used. Other more standard hardware components used would be a soundcard and speakers (if required), and a monitor. The quality of these components being closely linked to the potential quality of the VE experience provided.

1.4 Aims and objectives

The overall aim of this thesis is to develop structure and guidance for VE development by looking at the application and user requirements in conjunction with the technical options and constraints. This will be done through the identification and examination of the developer and user issues raised through a number of in depth case studies, the proposal of VE development strategies, and the evaluation of VEs built using those strategies. This structure and guidance should help to; improve VE utility, reduce time to build, and increase the performance of the VE with respect to functionality and usability.

Specifically, the objectives of this thesis are;

1. To use a case study based approach to establish the issues that are most relevant to the VE developer.
2. To use the methods and techniques learned from these case studies to define models for the various aspects of VE development, such that the process can be more easily understood and thus improved.
3. To review the existing Virtual Environment Development Structure (VEDS) through its application during VE development case studies.
4. To use the results of this review, together with the models previously developed, as a basis for the formulation of new sections to be added to the structure for the development of VEs.
5. To identify parts of the VE development process where extra guidance is required in order to avoid VE design problems.
6. To create exemplar VE development guidance tools to help with specific aspects of the VE development process.

This work will involve;

- Describing the technical limitations and trade-offs involved in VE development.
- Examining the users' experience in a VE and the issue of VE usability (as against VR system usability), and relating this to the VE development process.
- Defining a VE development structure.
- Illustrating the need for guidance through case studies.
- Producing guidance for the VE builder that enables them to get more from the VR system, minimising the technical limitations and maximising the user's experience.

Within these objectives, questions addressed by this thesis are;

- How can VE development efficiency be improved?
- What are the requirements for successful VE development?
- How should the components of VEs be categorised?
- What is the sequence of VE development (what activities are involved)?
- How can a VE be developed to match the expected users' requirements?
- How can a VE be developed to achieve its purpose (match the client's requirements)?
- What should be included in a VE (and what can be left out)?
- What type of tasks can be effectively modelled in VR?
- Do VE developers need guidance?
- What form should VE development guidance take (e.g. rules, hints, ideas, concepts)?
- How realistic should elements in a VE look?
- How realistically should a VE behave?
- How closely should the users' actions in a VE match the equivalent actions in the real world (what metaphors should be used)?
- What facilities should be provided outside or on top of the 3D display (e.g. buttons, text boxes etc.)?

Chapter 2 A Review of VE Development Issues

2.1 Introduction

The first part of this chapter is an overview of the chronology of VE/VR evolution with particular emphasis on the issues of this thesis, namely structured VE development, usability and functionality. Later in the chapter, topics emerging from the chronological review will be discussed in more depth along with other issues from the literature.

2.2 Time line of VE/VR evolution

The first publication in which an author theorises about the concept of a computer-generated environment was Ivan Sutherland's 'The Ultimate display'. He suggested that in the future it would be possible for a computer interface to simulate an artificial space in which the user's position and movements could be tracked (Sutherland, 1965). EUROGRAPHICS, the European Association for Computer Graphics, was formally constituted in spring 1980, and their first international conference was held in Geneva that September. The first time the general public was exposed to VR was in 1982 with the release of the movie 'Tron'. It was a sort of adventure in cyberspace (a word not in use at the time) in which a man gets trapped in a 3D computer game. The first 3D wire-frame computer games (such as Battlezone) had appeared in amusement arcades about a year earlier. In 1983 Myron Kreuger's book 'Artificial Reality' was published (Krueger, 1983), in which he made serious proposals as to how the technology could be developed and implemented. 1984 saw William Gibson's work of sci-fi fiction, 'Neuromancer', in which he coined the term cyberspace (Gibson, 1984). In 1989 Jaron Lanier gave interviews to the New York Times and other publications, using the term Virtual Reality for the first time. The beginning of the 90s saw PCs becoming powerful enough to render simple 3D environments in real time. There was now the potential for the development of interactive VEs on desktop PCs, and Dimension International (later to become Superscape) exploited this potential with the release of their Virtual Reality Toolkit in 1991. The first Virtual Reality International Symposium (VRAIS) was held in Seattle in September 1993.

The first serious attempt to tackle VE usability and structured design guidance to achieve this came about in 1994 when a 'Workshop on the challenges of 3D Interaction' was held (Herndon et al., 1994). Discussions were held on the following topics:

- Application space
- Fundamentals of 3D interaction
- Psychology (perception, and evaluation of user interfaces)
- Conceptual design (user interface design)
- Current state of the art in 3D user interface research
- Non-traditional interfaces

Many ideas were put forward, few conclusions were reached, but for the first time questions were asked that would instigate the appropriate research. Recognising the need to give structure and definition in order to increase understanding of the VE development process, the group with which the present author is associated at the University of Nottingham published the first version of their 'framework for the development of VEs' in 1996 (Wilson, Cobb, D'Cruz, & Eastgate, 1996).

Dissatisfaction with the term 'Virtual Reality' had been building up for some time within the VE/VR community. It was a term that promised more than the technology could deliver and distracted from the technology's more practical and achievable applications. John Wann summed it up well in 1996 when he said *'Virtual Reality is an oxymoron that is misleading and unnecessary'* (Wann & Mon-Williams, 1996). This was backed up the following year by Davis when he said *'The construction of virtual environments is best seen as the construction of meaningful forms and experiences rather than as replication of the real world'* (Davis & Athoussaki, 1997).

In 1997 the EPSRC started funding the INQUISITIVE (INcreasing Quality of User Interfaces for Strategic Interactive Tasks In Virtual Environments) research project. The collaboration between the Computer Science Department at the University of York and the Rutherford Appleton Laboratory was set up to look at the design of user interfaces for virtual environments, particularly in the domains of training and simulation. The group are working towards the development of an interaction toolkit that will allow portability and consistency across different platforms. Initially this toolkit will be developed for use with the dVise and Maverick systems (Boyd & Sastry, 1999).

The first published attempt to give guidance to the VE development process came in 1997 with the publication of the 'Taxonomy of usability characteristics in Virtual Environments' (Gabbard & Hix, 1997). At nearly 200 pages this covered the topics in some depth, but still required developers to go in search of other sources to find the specific usability guidance suggested. In the following year Kulwinder Kaur published her doctoral thesis in which she presented her more self-contained but less comprehensive 'Design advice tool for presenting usability guidance to VE designers' (Kaur, 1998).

In December 1998 a workshop was held at De Montfort University in Leicester, UK, with the title "The First International Workshop on Usability Evaluation for Virtual Environments" (abbreviated to UEVE'98) (Tromp & Fraser, 1998). This workshop targeted VE user studies such as VE interaction, navigation, social interaction, presence, general utility, and methods to perform VE user studies such as controlled experiments, user observation, user reports, interface inspection, and design guidance.

In September 1999, the University of York hosted a workshop entitled " User Centred Design and the Implementation of Virtual Environments". The aim of this workshop was to provide a forum for discussing the development of VE solutions that are appropriate to users' tasks and requirements. The presentations covered the areas of modelling and design of virtual environments, toolkit design for effective interaction, and problems and issues involved in moving from virtual environment design to implementation.

Gabbard (1999) highlighted the continuing VE design problems by asserting that *'The vast majority of VE research and design effort has gone into the development of visual quality and rendering efficiency. As a result, many visually compelling VEs are difficult to use and thus unproductive.'* - and that - *'Very few experts exist in user interaction design and evaluation of VEs.'* (page 51). Two years later and the problems still persist according to Fencott (In preparation) in which he states *'Our understanding of VR as a communications medium is not as well developed as the technologies of VR themselves. Thus our ability to construct effective, user centred VEs is still very much reliant on individual knowledge coupled with prototyping and incremental development. The problem with such knowledge is that it is not generic and does not easily allow us to apply it to other applications areas ...'* .

2.3 Formalised design process - Approaches to VE design as a whole

Over the last decade there has been much research into the development of toolkits to facilitate the construction of VEs (e.g. DIVE, AVIARY (Snowden, West, & Howard, 1993), MASSIVE (Greenhalgh, 1997), SVE (Kessler, Bowman, & Hodges, 2000)) but little research into how best to use these toolkits to build VEs that are usable, and even less into how to implement the functionality required to give VEs the required utility. Much research has been carried out to find low level solutions to VR system problems (such as techniques to reduce VR system latency (Reddy, 1997)), and some research has been done at the top level, looking at how the components of the VE should be brought together to form a coherent virtual experience, e.g. (Tromp & Fraser, 1998), (Fencott, In preparation). Low level research which could have great significance to VE developers and VE usability is in the area of providing cross platform toolkits for the construction of VEs (such as Boyd et al (1999)). This would allow VE developers to develop applications with a consistent user interface regardless of the operating system, a facility already available to those building 2D WIMP (widows, icons, mice and pointers) based applications.

2.3.1 Categorisations and Frameworks

In a book chapter investigating the ways in which the user cognitively interacts with a VE, Wickens (1995) suggests that partitioning VE systems into components gives the designer more flexibility to configure a VE in a way that is most appropriate for its use. This could allow designers to minimise time spent on less important components or omit them altogether. He also finds that full fidelity is unnecessary and costly (in terms of the consequences of rendering at full fidelity on system performance) and that an analysis of the tasks performed by the user can lead to a decision as to which features need to be modelled in the VE.

Ferwerda et al in Herndon (1994) concluded that the characteristics that 3D user interfaces must have to exploit the perceptual and reasoning skills of users fall into five categories: functional fidelity (as against full fidelity, i.e. sufficient information should be provided by the VE for user interaction to be successful), responsiveness (reduction of lag), affordances (clues about how to interact with the VE), appeal to mental

representations (making systems modelled within the VE appear and function in a way that is familiar to the user), and multiple/integrated input and output modalities (using more than just the visual channel for communication).

From their experience of developing VEs for industry, medicine and education, Wilson et al (1996) published the first version of their 'structured framework for building and testing virtual environments'. This framework attempts to give structure to the entire VE development process, from establishing the domain requirements through to evaluating the finished VE application(s). This framework has subsequently undergone further iterative development (D'Cruz, 1999; Wilson, 1997; Wilson, Eastgate, & D'Cruz, 2001) and is now referred to as VEDS, the Virtual Environment Development Structure. The developments that have taken place have added detail to the framework; for example the section on evaluation has been significantly expanded. The section covering VE building has yet to be expanded to give the detail required to inform that part of the VE development process. One of the aims if this thesis is the expansion of this section to show how the activities of 3D modelling, world assembly and programming functionality can be best applied to give the resulting VEs utility and usability.

Gabbard et al (Gabbard & Hix, 1997) have developed a framework that provides usability design guidelines to aid VE developers in specific situations. This work attempts to bring together the vast existing usability research in human computer interaction generally, and apply it to VE design, whilst recognising that issues such as presence and realism are extra factors when designing for VR/VE. Gabbard et al offer 195 guidelines on VE design issues such as locomotion, object selection and manipulation, user goals, fidelity of imagery, VE context, the use of metaphors, the use of agents (computer controlled characters), as well as offering guidance on the physical aspects of the VR system. As mentioned during the chronological review earlier in this chapter, in order to apply the guidelines appropriately Gabbard et al suggest that VE designers should follow the reference citations included in the framework to the literature source to find a fuller coverage of the specific issues. This may not be easy to do when one is part way through developing a VE. Further, multiple suggestions are often made, with the VE developer left to decide which advice is most suitable. As such these guidelines might be more appropriately used for finding solutions to problems arising during an evaluation stage of a VEs development.

In 1998 Kaur concluded *'There are no comprehensive methods or guidelines for considering user issues'* (Kaur, 1998). Kaur went someway towards improving this situation by developing a set of 46 guidelines in the form of 'design properties' that used a checklist approach to predict VE usability problems and suggest generic solutions. These design properties covered interaction in VEs, specifically the areas of the user task, spatial layout, viewpoint and user representation, objects, system initiated behaviour, actions and action feedback. A hypertext design tool was developed incorporating 12 of the guidelines and in a limited evaluation it showed good results in aiding object design and the incorporation of cues. She concluded that more work was needed to give guidance on the implementation of appropriate feedback. Taken individually, each of the guidelines is well researched and gives good advice about what cues are required in order to allow the user to understand and interact with a VE. However, the tool as it stands is very limited and if it were extended to include all aspects of VE development it could become unwieldy and difficult to use. This may be alleviated by a more rigorous use of the capabilities of the hypertext system giving more cross-referencing, none of which is incorporated in the current version. Also the tool has no flow between the various sections and therefore does not guide the VE developer through the process. This leads the present author to the conclusion that, as with the Gabbard and Hix Taxonomy (Gabbard & Hix, 1997) this tool could also be more suitably used during the evaluation stage of VE development.

In discussions at the First International Workshop on Usability Evaluation for Virtual Environments (UEVE'98) (Kaur Deol, Steed, Hand, Istance, & Tromp, 2000), several insightful observations were made.

- The (VE) design process needs to be understood before proposing guidance.
- Guidance would need to be quick and flexible to apply, and informal to complement design practice.
- The reasons for the importance of applying the guidance should be included to provide motivation to designers.
- Current successful games and VEs could be investigated to accumulate (VE development) techniques.

- Standards for VEs would reduce the learning needed to use a VE and allow users to immediately recognise features, but could be restrictive on design and result in less interesting VEs.
- User centred design should be encouraged.
- Usefulness and relevance of a VE application to an organisation should not be overlooked.

It is the present author's opinion that the need to understand the VE design process, and broader than that the entire VE development process, is the largest obstacle to the efficient creation of successful VEs. Much of the research documented in this chapter has looked into specific aspects of VE design, but with the exception of Wilson et al (Wilson et al., 1996) (Wilson et al., 2001) none have looked at the VE development process as a whole. If it is to be comprehensive it is hard to see how the guidance required can be made to be quick to apply, flexible and informal, although it is easy to see that these attributes are necessary for the guidance to be useful in a practical situation. Including the reasons for the importance of applying the guidance would tend to make the guidance more cumbersome in use. As in other domains where the results of design processes are released into the public domain (such as the car industry), VE developers already review each other's work and incorporate 'good ideas' into their own VEs. As many VE developers are also interested in 3D computer games they often borrow ideas from the games sector as well. However if VE designers do not understand the processes that lead to the efficient development of successful VEs they may be borrowing the wrong ideas and techniques (such as ones that are suited to a different user population, or ones that look 'cool' but have little practical value). Saying that 'the usefulness and relevance of a VE should not be overlooked' may be understating the case. Making a usable environment is only part of the story, a VE must be able to deliver at least some of the features laid down in the VE specification in order to be useful. A user centred design approach can be a powerful way of ensuring that a VE is both usable by, and has utility for, its intended user population (Neale, Cobb, & Wilson, 2000) (Cobb, Neale, Crosier, & Wilson, 2000).

The guidance tools that do exist for usability seemed to be concentrated on affordances and cues, with little discussion of the provision of feedback or broader issues such as navigation. The use of natural cues, language, symbolic signs (e.g. no entry).

highlighting, animations (e.g. to demonstrate actions), or using agents (e.g. to show the user around) were ideas suggested during discussions at UEVE'98. Against these suggestions the naturalness of the VE should be maintained and a balance should be reached between level of detail and runtime performance. When discussing traditional HCI guidelines, it was concluded that whilst some were relevant they needed adapting before they could be applied to VEs (e.g. because they don't take into account the VE requirement of maintaining naturalness).

Smith (2000) observed that one problem confronting VE designers is that the object geometry is designed and built before the object behaviour is considered. This results in many of the objects being built with too much or too little detail for the intended functionality, leading to time consuming remodelling of objects. He suggests that this is the result of the bottom up, technology led approach used to develop VEs, and that a top down solution would be more appropriate. His approach was to draw up a tree diagram showing a task based structural decomposition of objects, allowing a more comprehensive understanding of the issues to be tackled, before any geometry modelling takes place.

In recent work, ideas from the computer games sector have been combined with theories from the field of media studies to develop a Perceptual Opportunities (PO) model to help guide the overall design of VEs (Fencott, In preparation). This model categorises VE features into sureties, surprises and shocks. Sureties are features one would expect to find in a specific environmental context, such as furniture or trees. Surprises are unexpected features that are plausible and/or beneficial. Shocks are unexpected features that do not benefit the VE but rather draw attention to the limitations of the implementation of the technology and lead to a decrease in presence (e.g. rendering problems). Surprises are further divided into three categories; attractors, connectors and retainers. Attractors are features that encourage the user towards areas of interest, such as an animated object that can be seen from a distance. Connectors enable the user to follow a course between attractors and take the form of axes (paths), choice points (junctions) and deflectors (designed to inhibit certain activities). Retainers are major points of interaction designed to deliver specific objectives of the VE (the nature of which will be defined by the purpose of the VE). During the VE design process a perceptual map of these surprises can be drawn up that connects the various attractors, connectors and retainers to form a coherent virtual experience. The actual VE can then be developed from this map. Whilst

the present author considers some of the ideas to be flawed (for example, some connectors are not going to be surprises) it can be seen that the use of this method could lead to the creation of VEs where the activities are more integrated and coherent. It does not, however, deal with the detail design of the activities themselves.

2.3.2 User-centred and participatory design of VEs

User-centred design employs the users during the testing or evaluation phase of the development process, whereas in participatory design the users play the role of design partners (Crosier, 2000). In VE development the process adopted is often half way between user-centred and participatory with the users being involved at some level throughout. In her work using a user-centred design approach for the development of a Virtual City to teach life skills to children and adults with learning disabilities, Neale (1998) observed that *"by involving users from the first stages of product development, user abilities became apparent early on and usability difficulties were minimised"* (page 110). This should be equally true outside the special needs sector as there can be significant individual differences within a VE user population especially with respect to spatial abilities (Istance & Hand, 1998). When developing a VE science teaching application for mainstream education, Crosier (2000) concluded that an iterative user-centred design methodology is a suitable method for creating useful VEs that can be successfully integrated into schools. Outside the education sector Gabbard et al (1999) successfully employed user-centred design techniques in the development of a real-time battlefield visualisation VE, concluding that the methodology had a major impact in developing a VE to be both visually compelling and useful for solving real world problems.

2.3.3 Functionality

There is some reference in the literature as to the importance of using appropriate functionality in VEs, for example, in order to exploit real world experience (Gabbard & Hix, 1997), (Mason, 1996) but little on how to implement it. Kaur (1998) advocates a need for (VR development) toolkits with better facilities for modelling complex behaviours and suggests that it should be easier to comprehend functionality in a VE (than in a GUI), as it will be more naturalistic and less symbolic. This may be true if it is possible to model the functionality in a comprehensible way, given that the functionality being modelled may not be easy to understand in the real world. Object manipulation in

VEs is a subject often researched and there have been a number of publications suggesting methods of implementation (e.g. (Poupyrev, Weghorst, Billinghurst, & Ichikawa, 1998) and (Mine, Brooks, & Sequin, 1997)), but this research invariably concentrates on methods by which the user can use the input device(s) to directly interact with virtual objects, picking them up, carrying them, and placing them for example. Other work has been carried out developing real time algorithms for constraint-based modelling of collisions between 3D objects (e.g. (Fernando, Fa, Dew, & Munlin, 1995) and (Fernando, Murray, Tan, & Wimalaratne, 1999)). Very little research has been done into the higher-level development required to implement functionality within a VR toolkit (such as Superscape or World Toolkit), with a view to modelling the manipulation of virtual objects using semi or fully autonomous processes such as virtual cranes or robots. The issues involved are ones of compromising between the programming demands of the functionality itself and the rendering demands of visually representing the VE including the functionality in real time using the available computing platform (Eastgate & Wilson, 1994). Eastgate et al used the examples of conveyer belts, forklift trucks and cranes to show how a level of functionality can be implemented. As well as trade-offs between functionality and visual detail, specific issues encountered were;

- Modelling of physical properties
- Object boundaries and collisions
- Choosing between simulation and representation
- Object hierarchies
- Modelling large numbers of dynamic objects
- Modelling interactions with, and between autonomous and/or semi-autonomous processes
- Movement of objects over non-planar surfaces

This implementation of functionality was explored again in Eastgate (1997) and Wilson (2001).

2.3.4 Relevant research from other disciplines

From the field of usability engineering, Nielson (1993) describes usability as having the characteristics of being easy to learn, efficient to use, easy to remember, causing few

errors, and subjectively pleasing. Whilst Preece (1994), describes usability as having a number of factors including the interface, tasks and hardware, the needs, capabilities and characteristics of the user, and the context within which a system is used. These definitions of usability can be applied to VEs. However, despite the fact that there is much design process guidance for human-machine system or human-computer interaction, (e.g. (Newman & Lamming, 1995), (Sanders & McCormick, 1993). (Sutcliffe, 1995), (Wickens, Gordan, & Liu, 1998)) it has been frequently observed that this guidance needs to be adapted before it can be applied to VEs (e.g. (Gabbard et al., 1999), Isaacs in (Herndon et al., 1994), (Kaur Deol et al., 2000), (Wilson et al., 2001)). As Gabbard states, *"limitations and incompatibilities between GUIs and VEs may render these methods inapplicable at best"*. Further, the guidelines and models within the human factors and "conventional" HCI communities have been criticised for being not always useful or usable by designers, engineers or even ergonomists (Wilson et al., 2001). Schaaf Jr. (1998), as well as pointing out the differences between techniques required to evaluate VEs and those used in other disciplines, goes further by arguing that there is a need to make distinctions between various VEs themselves. For example transference is of prime importance in a VE used for training but is of little importance in a VE used for data analysis. These differences between VEs may well extend to usability for example, where one VE consists primarily of navigational activities but another has mostly object interaction.

It is hard to draw upon any similar development guidance in simulation design because the range of technologies that fall within the definition of simulation is so diverse. At one extreme are the flight (and similar) simulators, where the actual controls of the system are physically built into the simulator as a 'mock-up', the experience is more real than any VR system and the usability issues are the same or very similar to those for the real system being modelled. At the other extreme are modelling systems that are command line driven with text output, where no attempt at sensory realism is made. VEs have sufficiently different attributes and purpose for their own structured development framework and associated guidelines to be needed. Specifically these attributes are the generic nature of the input and output devices (not designed specifically for an application as they would be, for example, for a flight simulator). and the naturalistic, 3D, interactive nature of the VE database itself. The purposes for which VEs are developed are much broader than the training or system testing that simulators are usually used for.

Of course there is significant overlap between VR and simulator technologies, both in terms of technology and application. In fact the development of VR technology has made personal simulators practical and affordable for everyday use (Ellis, 1995). Non-graphical simulators can have a VR capability added to make the user interface more naturalistic, or to give it a 'man-in-the-loop' capability (e.g. (Eyles, 1991)). but many of these systems just have a graphical display of the simulation without giving any facility for naturalistic interaction (Ellis, 1995). Alternatively, VR/VE technology can be used to replace the expensive hardware interface of a 'mock-up' type simulator (e.g. (Wirth, Sokolewicz, Bohm, & John, 1995)).

From the field of computer games an initiative has been started to develop a set of Formal Abstract Design Tools (FADT) (Church, 1999), to give structure to the user's experience of a game. These tools attempt to group features of games into categories such as (user) intention, perceivable consequence, and story. By building up this terminology, also known as the Game Design Lexicon, it is hoped to facilitate collaborative design and analysis, leading to better games and more satisfied users. Church has set up a forum on the www.gamasutra.com website where game designers can suggest terms for the lexicon and discuss them with other games designers. He hopes that ultimately, the currently typical criticism and discussion of games (such as whether a game is 'fun') can be replaced by more precise and effective communication.

2.4 Navigation

User problems when navigating in VEs were summed up by Ruddle (1998) as being users' lack of knowledge of their position, their orientation and a VE's structure, and a general lack of familiarity with using VEs. As a result of an experimental study looking into the use of a virtual compass as a navigational aid, he concluded that desktop VR may cause specific navigation problems due to the fact that the user does not physically turn their body when they change direction in a desktop VE (as they would in the real world, in an HMD or in a cave). Of the 4 user problems identified by Ruddle, the first three are factors in navigating in the real world (replace the phrase 'a VE's structure' with one appropriate for the real world such as 'the environment's structure'). His terminology for the fourth user problem he identified, 'the lack of familiarity with using VEs' does not go far enough to describe all the problems users have with the hardware input devices, the various metaphors used to translate manipulation of the input devices into actual movement within the VE, the narrow field of view provided by most display devices (caves and reality theatres excepted), the monoscopic nature of most display devices (stereo HMDs excepted), the low level visual fidelity provided by most VR systems, and the lack of other sensory feedback (e.g. 3D audio).

Darken and Sibert (1996) compared different types of VE with varying levels of superimposed grids and maps to find out what effect this had on way-finding behaviour (purposeful, oriented movement during navigation). Whilst concluding that more work needed to be done they found that in general devices as simple as reference points improve navigation. In later work they recommend that VEs should be given an explicit structure, dividing the world into small distinct parts, to allow the user to mentally organise the environment (R. P. Darken & W. P. Banker, 1998). Other research looking into the use of maps as VE navigation devices to 'speed up the process of spatial knowledge acquisition' was carried out by Istance (1998), who argued that more attention should be paid to the individual characteristics of users with respect to their navigational abilities.

In their journal paper of 1998, Waller et al (1998) document a series of experiments comparing VEs with 2D maps as methods of transferring spatial knowledge. Whilst their results were not conclusive, their research went some way towards conceptualising VE

navigation by dividing it up into interface (input and output devices) fidelity and environmental (the VE itself) fidelity. This emphasis on fidelity as against usability reflects the fact that they were interested in the use of VEs as a direct replacement for a real environment without using the capabilities of VR technology to enhance environments to improve usability (such as adding extra cues to aid navigation or interaction).

Neale et al (2000) suggest a number of ways in which the design of the VE can be adapted to aid navigation especially for users with (real world) disabilities. These methods include adding more cues such as arrows and signs to indicate the direction to go to reach a specific target, and changing the layout of a VE to make reaching the target easier, e.g. by increasing the size of corridors and doorways. These techniques have advantages for VE developers in that they will not significantly increase the development time of a VE in the way that providing a compass and/or maps could.

The methods of aiding navigation can be summarised into three groups; providing a compass and/or some form of map of a VE, providing better cues as to the location of features within a VE, and changing the structure or geometry of a VE to aid navigation. Each of these methods has its merits but their appropriate use will depend on the nature, size and complexity of the VE and the characteristics of the user population. Further conceptualising navigation in VEs could increase understanding of the subject and lead to more structured approaches to its implementation.

2.5 Interface tools/metaphors

In the introduction to his paper reporting on the Workshop on the Challenges of 3D Interaction, Herndon stated that *'The most effective ways for humans to interact with synthetic 3D environments are still not clear'* (Herndon et al., 1994). Later in the same report, in a section on the fundamentals of 3D interaction, Mackinlay and Kettner (1994) write that *'The interaction techniques used in today's 3D graphics applications to manipulate synthetic objects and navigate through synthetic worlds are generally ad hoc implementations of task dependant designs. Little work has been done to unify the wide variety of techniques into a universally applicable set. There is, however, a feeling that a general set of 3D interaction techniques will eventually emerge'*. On the other hand, Isaacs et al (1994), in a section on user interface design in the same paper, contend that user interfaces to 3D graphics applications must be tailored to suit their particular user communities (architects, surgeons, designers etc, also novice or expert user of 3D graphics interfaces). On the subject of metaphors, Mackinlay and Kettner write that *'Metaphors are something of a mixed blessing, however. For instance, if a particular metaphor is interpreted literally by users, they may expect that their real world knowledge of that thing will transfer into the synthetic world and that the interface will thus require little learning. In all likelihood however, it will not transfer in full, since metaphors are rarely implemented completely*' (page 3).

In 1995, in the introduction to their book 'Virtual Environments and Advanced Interface Design', Barfield and Furness write about the attributes of an ideal medium, then go on to look at the shortfalls in the current (non-VR) interfaces and how VEs can reduce but not eliminate these shortfalls (Barfield & Furness III, 1995). The limited field of view of the displays, the methods of manipulation with the current input devices, and limited use of 3D visual, acoustic and tactile displays are the main shortfalls highlighted. They suggest that these reduce the bandwidth of the flow of information between human and the machine, limiting the technology's capabilities as a *'tool to extend our intellectual reach'*. Finally they list some of the issues that need to be resolved with respect to virtual interfaces and VEs. These include;

- the *'need to develop a theoretical basis for the work done in VEs, and a need to develop conceptual models to assist the designers of virtual worlds'*.

- the *'need to develop a solid understanding of the human factors design implications of virtual interfaces'*.
- the *'need to develop languages, spatial and state representations, and interactive heuristics for constructing virtual worlds'*.

They conclude by stating that there is a *'need to design more natural and intuitive interfaces to virtual environments'*.

Recently, a significant amount of research has been done in America into developing interaction techniques and user interface designs for immersive VEs (Bowman, Johnson, & Hodges, 2001; Bowman, Kruijff, LaViola, & Poupyrev, 2001). This work was more concerned with ways of using the hardware input and output devices and less concerned with the design of the VE itself.

In a paper giving an overview of human factors issues in VEs, Stanney stated that *'Research into the design of new VE metaphors - and, more importantly, guidelines to develop such metaphors - is needed'* (Stanney, Mourant, & Kennedy, 1998). She goes on to suggest that the constancies, expectations, and constraints elicited from interactions in the real world should be designed into VE metaphors, and that this would potentially result in a more intuitive interface. However the present author is of the opinion that whilst interactions in the real world may have some constancy, it is the lack of consistency in real world interactions that makes it difficult to design a consistent and intuitive VE user interface. These interactions include those related to navigation (e.g. walking, jumping, sitting down), grasping and releasing objects, moving objects from one location to another or reorienting them, applying one object to another (e.g. hammering a nail), and activating a part of an object (e.g. switching on a piece of equipment). This list is by no means comprehensive and each of these types of interaction requires unique (so that the actions can be distinguished) metaphors that will also show wide variation in how closely they match the equivalent real world activity.

2.6 Discussion of the Literature Review

Many of the problems facing VE developers to date have not been conclusively addressed by the literature. Apart from the general problems of improving VE usability, particular problems are in the areas of designing VE functionality (especially interaction metaphors and feedback), and structuring the VE development process.

A partial solution to these problems can be found in the employment of a user-centred/participatory design methodology to tailor a VE to the (range of) usability characteristics of the user population, and to iron out any usability and utility problems shown up during evaluation. If the presently available, largely retrospective, guidance could be replaced by more proactive techniques this may significantly shorten development times by reducing the amount of VE revision required.

There is some disagreement in the literature about the implementation of interaction metaphors, with some advocating consistency (e.g. Mackinlay and Kettner in Herndon (1994)), others arguing more towards faithful representations of real world interactions where possible (e.g. Stanney (1998)), and others saying that the metaphors should be tailored to the needs of the expected user population (e.g. Isaacs et al in Herndon (1994)). It is the author's view that interaction metaphors need to be developed that form the best compromise between these three views, i.e. matching the equivalent real world interaction, being easy for the expected user population to carry out with the available input devices, and maintaining consistency with other VE interaction metaphors in the same VE (at least). The development of these metaphors will have to take into account the sophistication of the i/o devices available, the nature of the equivalent real world interaction, and the aptitude of the user to both the real task and the virtual representation of that task (including the i/o devices of the VR system itself). This discussion about metaphors can be extended to; whether to use non realistic elements in a VE, whether to position some of the interface outside the VE (e.g. a hybrid interface with buttons and icons), what level of realism should be built into a VE and what objects need to be included in a VE.

The VE development process needs to be given more structure. This can be done by defining and categorising the components of VE development and fitting them into a

framework that shows their sequence and the relationship between them. This structure should take into account the needs of the user as well as the specified purpose of the VE.

Whilst reviewing the literature for this chapter, it came to the notice of the present author that the multidisciplinary approach required to develop VEs has led to a situation where there is little shared terminology amongst developers and researchers from different groups. For example, a developer from a computer games development background may refer to a 'viewpoint', whereas a developer from a CAD background may refer to a 'camera'. As each researcher conceptualises their own understanding of the VE design process, another set of categorisations and theories, each with their basis in sound but differing human factors, computer science and/or multimedia research (etc), and each with their own set of acronyms, is added to a disparate pool of knowledge. In an article in Games Developer Magazine, Doug Church explains the need for *'a shared language of games design'* (Church, 1999). There is similarly a need for a shared language amongst VE developers to enable them to communicate more effectively and thus speed up the evolution of VE design. However, the present author recognises that a shared language cannot be implemented overnight, even if all the relevant people agreed on its desirability.

It may be that once the issue of VE development structure has been tackled, it will become possible to develop a form of guidance that can be used proactively, possibly in conjunction with a user-centred design methodology, to help VE developers develop useful and usable VEs and increase the efficiency of the VE development process.

2.7 How can we give Structure to the Development of VEs?

2.7.1 Interfaces in VR system use

In order to give structure to the VE development process one has first to define the context within which the VE developer is working. Figure 2.1 shows the author's view of the relationship between the user and the VR system on both the physical and cognitive levels. At the bottom we have the VR system platform, which comprises of the hardware and software necessary to process and render a real time 3D VE. At the top of the diagram, and conceptually sitting upon the VR platform, is the virtual experience. The

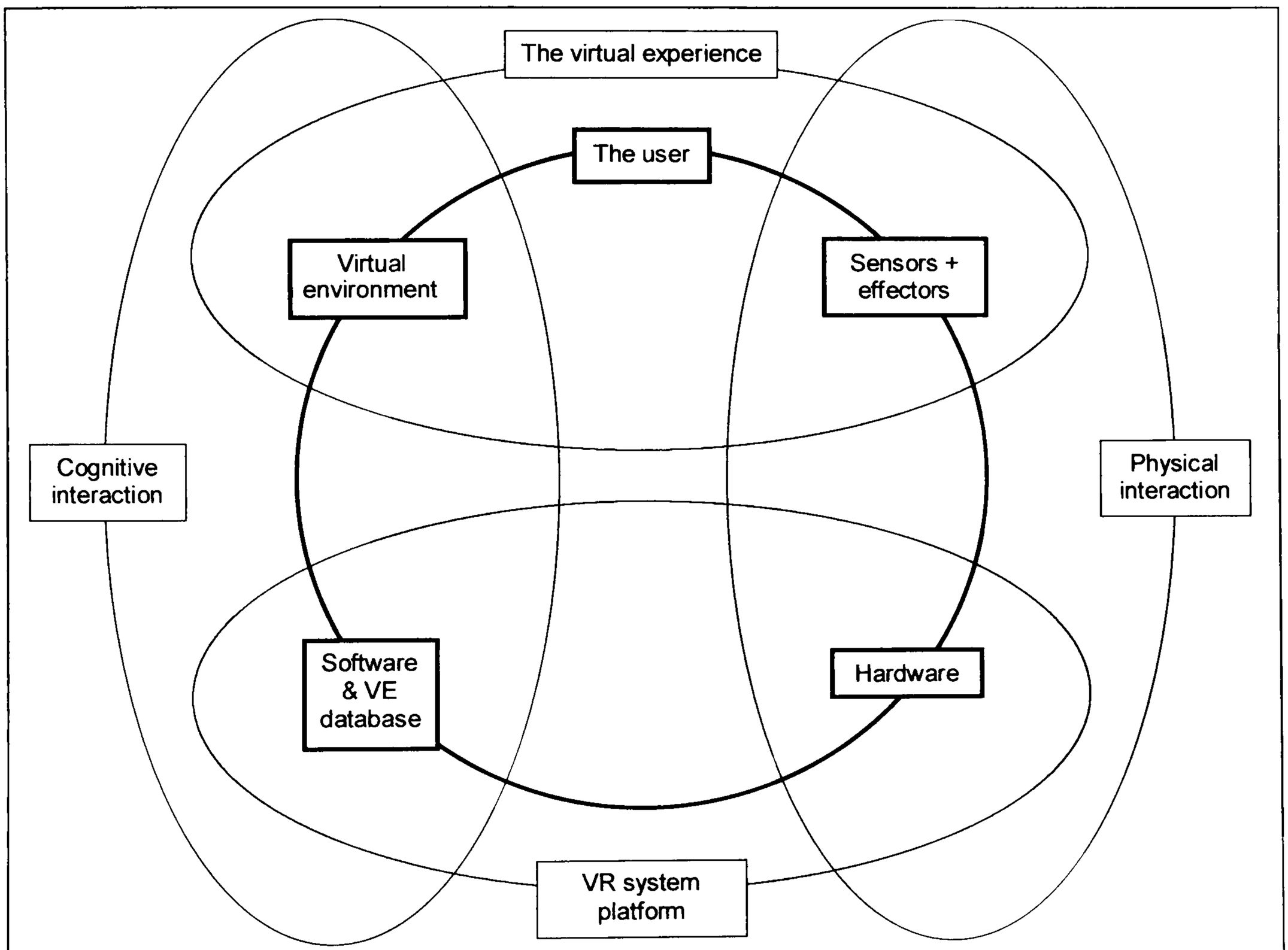


Figure 2.1. Interfaces in VR system use

user is within the virtual experience, and they are experiencing the 3D VE and the sensors and effectors that allow them to see (hear and feel) and interact with the VE. Physically, the user only interacts with the hardware sensors and effectors, there is no direct physical connection between the user and the 3D environment. The VE database is stored in the memory deep within the system unit. Cognitively however, the user should feel that they are actually moving through the 3D space and manipulating objects within that space, i.e.

that they have presence in the VE. The user will however, be aware that they are doing this via a synthetic interface, both in terms of the hardware and the software.

2.7.2 The four building blocks of virtual environments

In order to understand how a VE is built it is necessary to understand what the components of a VE are and how they relate to each other. There are many ways of describing the contents of a VE. One of the simplest is to say that a VE consists of four building blocks; topography, objects, behaviours and viewpoints. Here, the term topography is being used to refer to all aspects of the layout of the VE. The objects within the layout are made up of combinations of shapes, colours and textures. Some of these objects may have dynamic or transient characteristics, which can be described as behaviours. Whilst a VE can exist without any viewpoints, these have to be set up so that a user can experience the VE. These four building blocks are not processed in any fixed sequence. The specifics of the VE being developed will largely determine the order in which the work is done, but it is common for the process to involve frequently switching between object, topography, behaviours and viewpoint creation. It is therefore, in no particular order that these building blocks are described in the following sections.

VE topography

Generally speaking VEs have naturalistic topographies in which the real world characteristics of 3D space are reproduced. It is possible in a VE however, to create unnatural topographies such as 'portals' or 'teleports' to distant parts of the environment, or 'tardis' type structures, all of which are not currently possible in the real world (Greenhalgh et al., 2000). The topography of a VE is structured hierarchically (see figure 2.2) such that objects at the top of the hierarchy have a location and orientation relative to the VE as a whole. Objects lower down the hierarchy have locations and orientations relative to their parents. This means that if an object moves around the VE, its children will change location and orientation accordingly as if they are fixed to the parent. The implementation of the topography is usually the most straightforward of the four building blocks of VE creation.

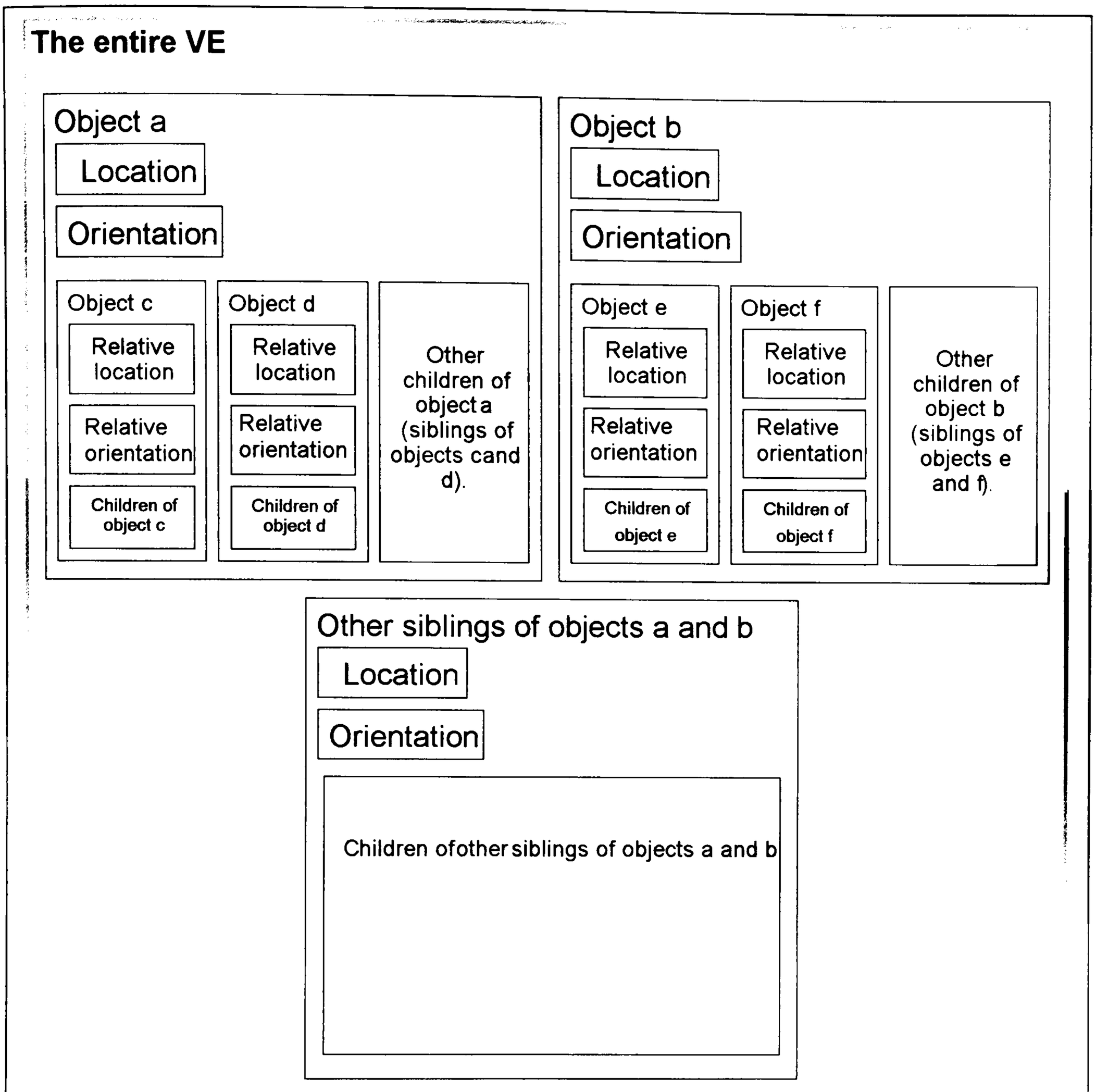


Figure 2.2. A logical model of the hierarchical structure of VE topography

Virtual objects

By referring back to the way these issues were tackled it is possible to start building a picture of how virtual objects are created. In their simplest form objects have shape, size and appearance. They may be made up of several child objects in which case they will have their own internal topography. They may also have behaviours and dynamics, which will be covered later in this section. The procedure for the creation of virtual objects is shown in figure 2.3. The shapes are created using a 3D modeller. Some VR toolkits (such as Superscape) have a shape editor built in. For other platforms an external 3D modeller will be required and the shapes created will have to be imported into the VR application. At this stage the shapes will consist of a number of flat facets. The number of facets used to create the shape has implications for the VE, particularly the rendering

speed (Eastgate & Wilson, 1994). The simplest way of rendering a shape is to give each facet a colour. More complex visual effects can be achieved by applying textures to the facets to create visually realistic objects. This also has implications for the VE similar to those for the number of facets used. Applying a gradual, smooth shading across several facets can simulate curved surfaces realistically, although the geometry is still made up of flat facets. Having given the shapes the required appearance they can now be assembled, if required, to make more complex objects. Earlier monitoring of the number of facets

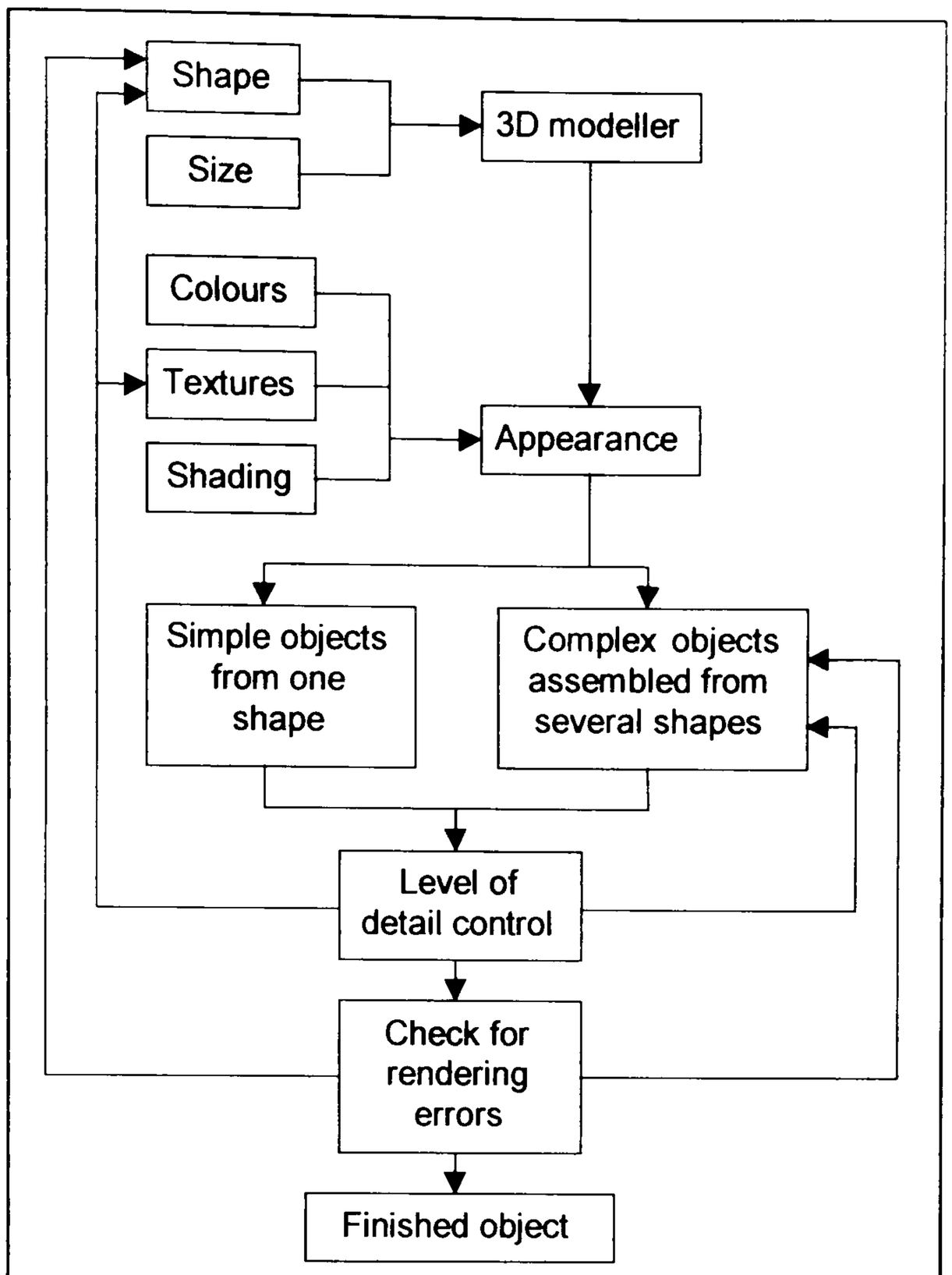


Figure 2.3. The procedure for creating a virtual object

used and the number and resolution of the textures used should help to produce an object that does not require too much processor power to render. However at this stage it is prudent to visually check the finished object looking for redundant facets or textures that are unnecessarily detailed. Different VR platforms have their own techniques for sorting the object facets for rendering. Some of these techniques are more efficient than others, all are fallible, and so it is important to check objects for rendering errors. These errors may be due to the way a shape has been constructed (e.g. the order in which the facets were added), or the way the shapes have been assembled to make an object (e.g. forming overlaps).

Object behaviours and dynamics

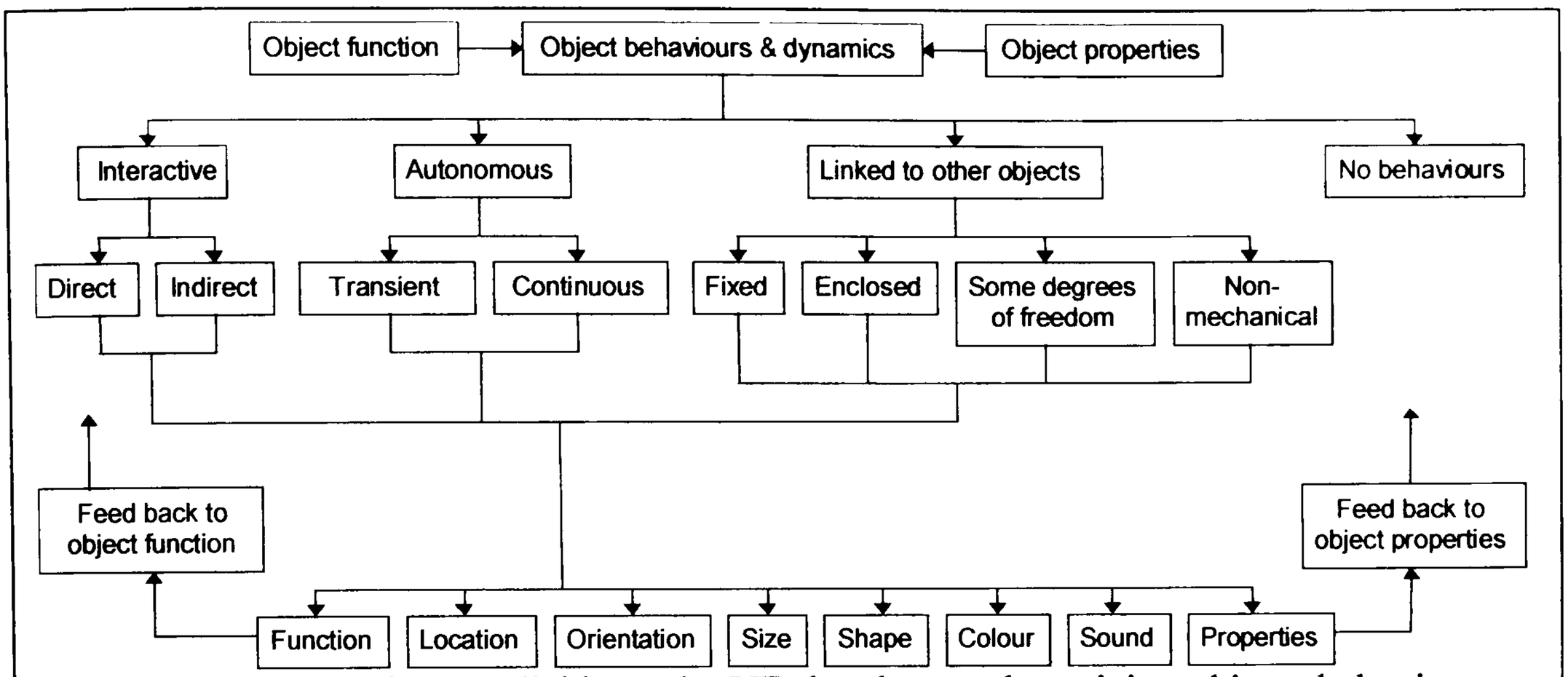


Figure 2.4. The choices available to the VE developer when giving objects behaviours and dynamics

Figure 2.4 shows how an object's behaviours and dynamics will be derived from its function and properties. These behaviours and dynamics can be categorised as interactive, autonomous, linked, or 'no behaviours' (these categories will be discussed in more depth later in this section), and manifest themselves as changes in location, orientation, size, shape, colour, sound emitted, function or properties. If a behaviour results in a change in function or properties of an object (e.g. a car runs out of fuel) this will result in a change in the object's behaviours and dynamics as represented by the feedback loop in the diagram. As explained in the previous section, building virtual objects gives them size, shape and appearance. In the real world, objects also have other inherent properties such as mass, centre of gravity and coefficients of restitution. To give all objects in a VE these and other inherent properties would be very time consuming and largely unnecessary. It is common for most objects in a VE to have size, shape and appearance as their only attributes. Other properties will only be assigned to objects where they are needed. In a VE it is also possible for an object to have unnatural inherent properties, such as being invisible, or being enterable whereby objects can enter or pass through another object.

As well as inherent properties, objects can also be given function, such as a door that opens when the handle is turned. These functions will often be in response to user interactions, but they could also be in response to the behaviours of other objects in the

VE, or entirely autonomous (e.g. performing a certain function after a pre-determined amount of time has elapsed). Sometimes the dividing line between object function and object properties is difficult to define. From the VE developer's viewpoint this is not important as these properties and functions will need to be programmed in similar ways and can all be considered under the umbrella terms of behaviours and dynamics.

The behaviours and dynamics of objects can be split into four types. The first group is interactive behaviours. These can be either direct or indirect, for example, when a light switch is switched from the 'off' position to 'on' position, the interaction with the switch is direct, but the interaction with the light, which changes its appearance, is indirect. The second group can be split into transient and continuous autonomous behaviours. An example of a continuous autonomous behaviour would be a virtual clock. An avatar performing a sequence of activities would be a transient autonomous behaviour. The third way in which virtual objects behave is in the ways in which they can be linked to other objects. This can best be explained by example. The windscreen is rigidly fixed to a car. The wheels on a car are fixed, but have one (rotational) degree of freedom. A second vehicle attached by a towrope has many more degrees of freedom but is still linked. The car passengers are enclosed by the car. An automatic garage door is not mechanically linked to the car but its behaviour is determined by the position of the car. The final category of 'no behaviours' is worth mentioning because it means distinctly different things in the real and virtual worlds. In a VE, an object that has no behaviours or dynamics assigned to it can hang in a fixed position and at a fixed angle in mid air with no means of support. In the real world this would be considered to be extremely strange behaviour.

There are many forms of object behaviour. They can change their location (or speed), orientation, shape, size or their colour (or texture). They can also change their function or properties (for example a door can become locked), in which case their behaviour and dynamics could take on a completely different type and form.

Viewpoints and navigation

A viewpoint onto a VE can either be attached to an object (such as a human form) or not attached to an object and therefore free to view the VE from any point and at any angle (see figure 2.5). Viewpoints can also be egocentric, i.e. looking from within an object, or exocentric, i.e. looking from a point not within an object. An exocentric viewpoint can still be attached to an object such that it moves with it, but views the world from a point not inside that object. Thus an exocentric viewpoint could be set up that tracks a moving object whilst viewing it from above (a birds-eye view). Alternatively an exocentric viewpoint can be set up that isn't attached to any object and can therefore have total freedom of movement, with the ability to pass through (non-enterable) solid objects such as walls. This type of viewpoint is sometimes referred to as 'ghost mode'. However, both attached and unattached viewpoints can be constrained by the number of degrees of freedom (DOF) they have. Any viewpoint can have DOF ranging from none, where the viewpoint is completely fixed, up to six (three angular and three linear, see figure 4.5), which allows the viewpoint complete freedom of movement. It is common for unattached viewpoints to have six DOF, or five where the facility to roll is removed as this can be confusing for users and is of limited use. The DOF of an attached viewpoint is the combined total of the DOF of movement of the object and the DOF of the attachment of the viewpoint to that object. For example the DOF of a railway train object on a straight

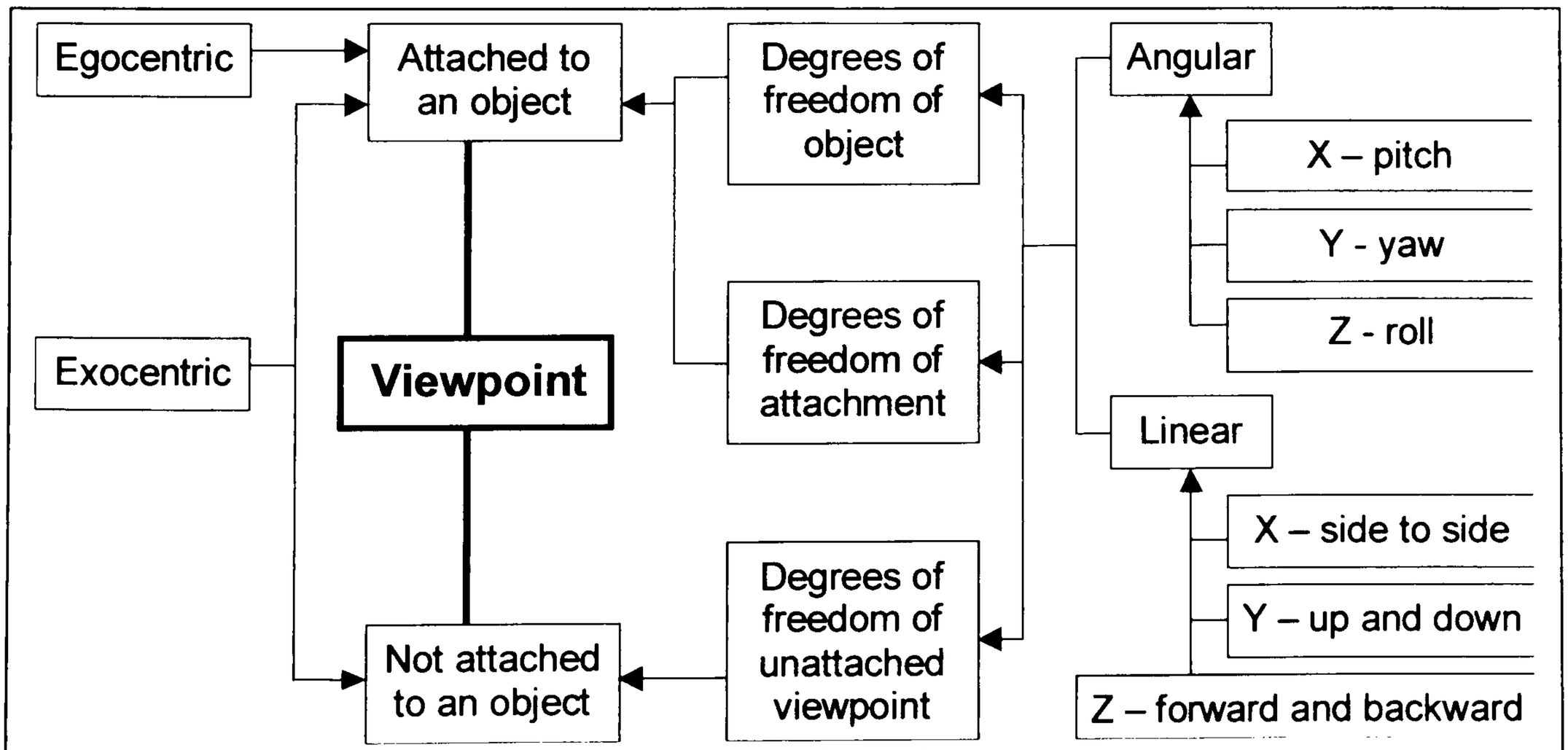


Figure 2.5. Types of viewpoint and their freedom of movement

track is limited to forward and backward (z-axis) movement. The attachment of the viewpoint to the train could in addition allow rotation about the vertical (y) axis (so that

the user can look at objects to the side of the track). The total DOF available to the viewpoint is therefore two, linear z-axis and rotational y-axis.

For a viewpoint to remain egocentric it must constrain the user to viewing the world in a way defined by the type of object that the viewpoint is attached to. The DOF of the object will be the result of setting up its behaviours as discussed in the previous section. Depending on the programming of the object, this may result in the constraint of the methods of movement and navigation available to those that would be possible by that type of object in the real world. For the viewpoint to remain egocentric, no linear DOF of attachment should be set up, as this would allow the viewpoint to move to a position outside the object. Angular DOF of attachment however, will not result in the viewpoint becoming exocentric (provided that the centre of rotation is correctly configured), but they may not be consistent with the behaviour of that type of object.

2.7.3 The VE experience

The four building blocks of virtual environments describe the contents of a VE but they do not explain how the user fits in. The section on interfaces in VR system use (figure 2.1) showed that the user interacts physically with the hard ware user interface, but

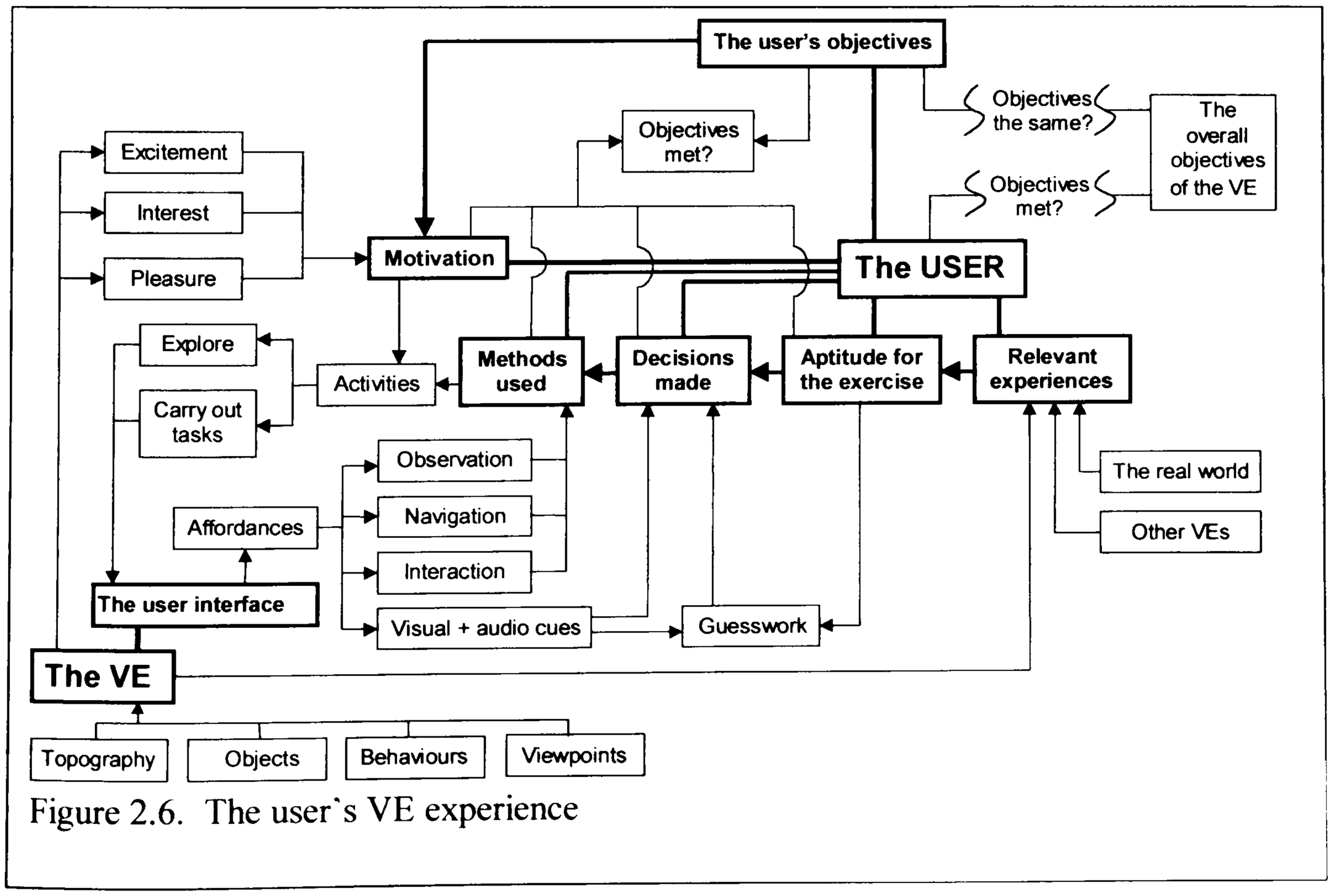


Figure 2.6. The user's VE experience

cognitively with the VE itself. In order to better develop VEs that are usable we need to look in more detail at the cognitive user interface and the way in which a user experiences a VE (figure 2.6).

Every user of a VE will be an individual with their own aptitude for the tasks and their own objectives. These characteristics will determine what activities a user performs, or attempts to perform in the VE, and the success of these activities may lead to the user's objectives being met. The VE developer however, does not work for the user, but rather for a client. This client may in fact be the user, but often the client will be the user's employer, or have some other relationship with the user. The client may have overall objectives for the VE, which differ from some, or all of the users' objectives. The job of the VE developer is to ensure that the client's overall objectives are met, but this can only be done by being mindful of the potential users' characteristics. The user's objectives will be met, firstly if the required facilities have been programmed into the VE, and secondly if the user's motivation and aptitude are strong enough to enable them to employ the correct methods to perform the required activities within the VE.

The user's aptitude for the tasks will partly be influenced by their relevant experiences in the real world and in VEs. This aptitude will affect the decisions made by the user when using the VE, which, in turn will determine the methods used to carry out the activities within the VE. However the user will only carry out activities where they are motivated to do so, and some of this motivation will come from the objectives they had when they entered the VE. The activities performed will largely consist of exploring and carrying out tasks, which the user does via the user interface.

The user interface will afford some visual and audio cues as to what might be possible in the VE. These cues will influence the decisions made by the user, some of which may be the result of guesswork based on previous experience as well as the cues provided. More fundamentally the user interface determines what means are available for the user to carry out the activities. These means commonly include observation, navigation and interaction.

The motivation to perform activities in the VE may be enhanced by the nature of the VE itself, giving the user excitement and pleasure and keeping them interested in the content

of the VE. As the user acquires experiences within the VE, these will help to increase their aptitude for the tasks and improve their subsequent decision making.

To summarise this from the developer's point of view; as well as understanding the reason for building the VE, the developer has to be aware of the characteristics of the expected user population. These characteristics include the users' previous relevant experience, their aptitude for the technology and the task, and their own personal objectives when using the system. Based on this understanding, enough stimulation, in the form of information, and opportunities for interaction, has to be provided to motivate the user to perform activities in the VE in order for the objectives of the VE to be met.

Chapter 3 The Application of VE Design Ideas

3.1 Introduction

In this chapter four case studies are presented. They are chosen from work carried out between 1992 and 1996, where the author was the principal developer. During the development of these VEs an attempt was made to increase understanding of the VE development issues. This involved the application of the models developed in the previous chapter, the further categorisation of VE development issues arising, and the development of new models representing VE development processes. This formalisation of the VE development process could then be used to develop new approaches to VE design. A summary of the case studies can be found in table 3.1.

	Project name	**Collaborating bodies**	**Description of application**	**Main design issues**
Case study 1	Virtual prototyping	Rapid Prototyping Research Group	Interactive 3D design and testing facility	Import/export of other file formats, object manipulation, user interface
Case study 2	The virtual factory	EPSRC, HSE	Exemplar VE used to demonstrate the possibilities afforded by VE technology to industrial users. Was also used to research the potential side effects of using VR technology	Context, cues and feedback, input devices, navigation, object manipulation, rendering speed
Case study 3	A virtual ATM	NCR	Training basic maintenance tasks on an ATM or cash-point	Context, cues and feedback, development time, focus vs. distraction, object manipulation, realism, rendering speed
Case study 4	PC network card replacement	EPSRC	Training a user to replace a PC network card	Context, cues and feedback, development time, focus vs. distraction, input devices, object manipulation, user interface, hybrid display

Table 3.1. A summary of the case studies examined during the exploration of development issues

3.2 Case Study One: Virtual Prototyping

3.2.1 Introduction

A close relationship between the Rapid Prototyping Research Group (RPRG) and VIRART at the University of Nottingham, gave us an opportunity to explore the feasibility and potential value of combining these two emerging technologies (Gibson, Brown, Cobb, & Eastgate, 1993; Wilson et al., 1995). Two main advantages of combining VR and Rapid Prototyping (RP) were anticipated. First, is the ability to interact with a virtual prototype in a way not afforded by a CAD model. The way a component behaves is a fundamental characteristic, and to be able to test this behaviour at the 3D model stage could shorten the design process. Secondly, is the ability to view and test a virtual prototype in an appropriate context. This could allow the designer to see whether a component fits physically, functionally and aesthetically with the rest of the system for which it is being designed.

It was apparent that, at that time (1992), VR technology was not capable of fulfilling a wide range of RP needs and by working with RP experts we first needed to establish what aspects of RP could be enhanced with the use of VR. We then needed to choose particular applications that could act as good initial feasibility studies. Assuming that the applications chosen would include some kind of real time design changes to a prototype, we would also need to develop a suitable tool to do this within the VE, or more specifically, within the Superscape Visualiser.

3.2.2 Choosing the applications

The first product chosen was a water thermostat housing from a car engine. This was chosen mainly for its familiarity as it had been used by the RPRG as the subject for many experiments in the past (Dickens, Cobb, Gibson, & Pridham, 1993). The real product would be conventionally made from a casting in one piece. The virtual model was built up from five shapes, grouped together, and given facet lighting to add depth to its appearance. The major need identified was for some sort of tool that allowed the model to be reformed rapidly, in (almost) real time, from its surface representation, thus providing a range of designs for assessment. These designs could potentially be assessed

for functionality, ease of maintenance, economy of material use, manufacturing and so on.

The second product chosen was the front panel of a personal computer system unit. This was chosen as an example where the switches, displays, and disc drives each had their own criteria for optimising their position and size within the layout of the panel. The components were made up of blocks which, along with blanking plates (used to cover unused disc drive bays) were all slotted together to make a model of the front panel. Other PC components (monitor keyboard and mouse) were then added to allow the design team to visualise the design in the context of the entire system.

3.2.3 Developing the Gyrotool

We needed a tool that would enable users to manipulate the virtual prototypes. This tool would need to have the facilities to:

- select object, section or area being manipulated
- choose type of manipulation (e.g. expand or move)
- choose direction of manipulation
- select coarse or fine manipulation (or step size)
- display current status

At that time we had not seen a suitable tool in any VR system. The Superscape system did not incorporate any object manipulation tools into its Visualiser; the Superscape World Editor was restricted to a user interface requiring numerical input of object size, position etc. The author had to devise a suitable user interface and make it work within the limitations of the Superscape Visualiser, and this tool had to be a true 3D device that would exploit the unique facilities offered by VR.

Objects within a Superscape VE are based on an orthogonal system of axes. Objects have their size and position defined by x, y, and z co-ordinates, so manipulation of these objects is also defined orthogonally. Also, manipulations to objects can only be made at the shape level or above in the Visualiser (point and facet manipulation can only take place in the Shape Editor), so objects that are to be manipulated need to be split into a sufficient number of shapes to permit the anticipated manipulations.

Based on these requirements and limitations we decided to base our tool on a system of x, y, z, axes which moves around with the user, maintaining a consistent orientation within the VE and a consistent position within the user's viewpoint (figure 3.1). At each end of each of the axes are buttons that always orient themselves towards the user. These are marked x, y, z, at the positive end of the axes, and are blank at the negative end. They are

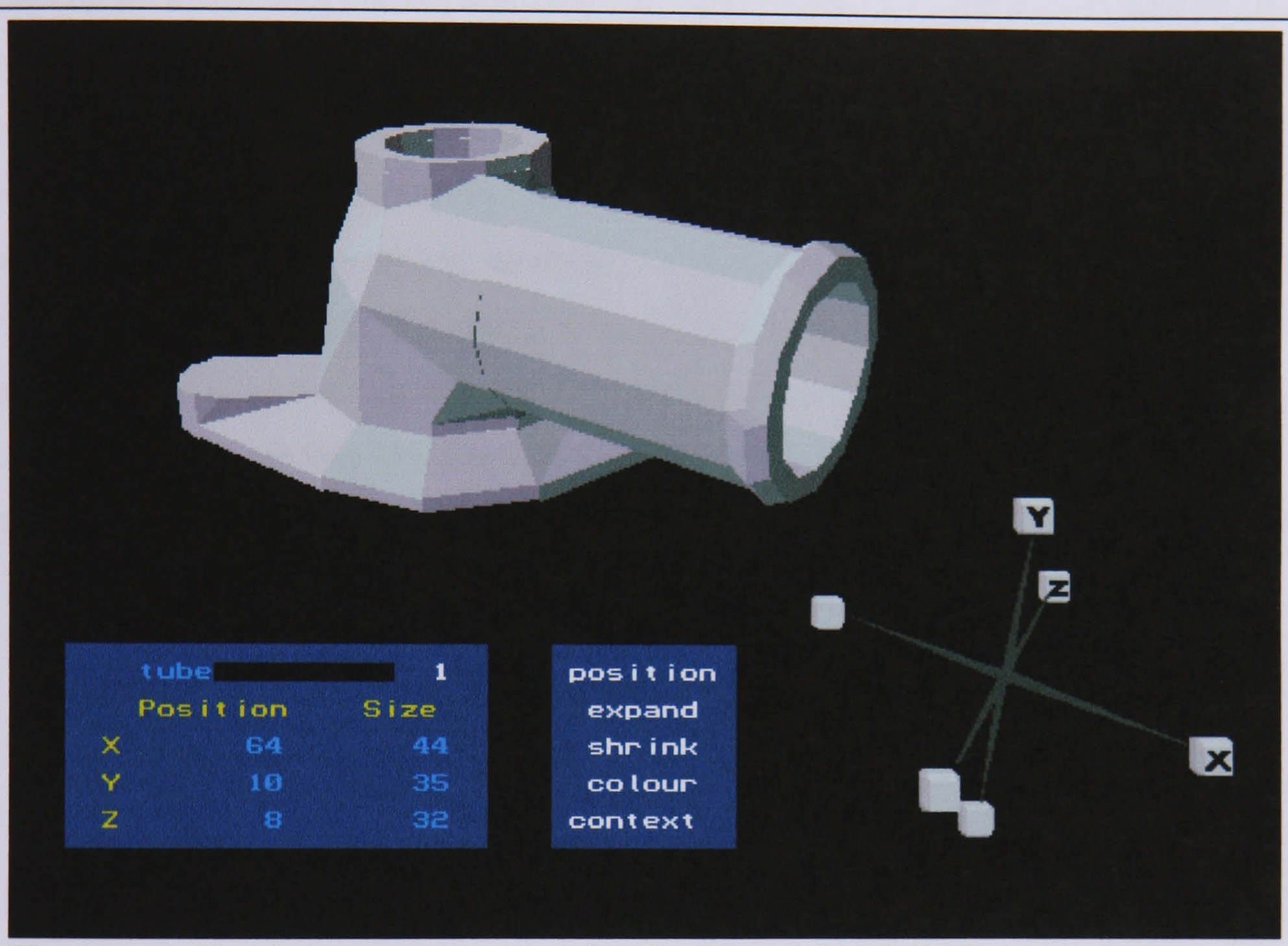


Figure 3.1. The thermostat housing with the Gyrotool and display

the main means of manipulating shapes within the VE. Because of this tool's ability to re-orient itself correctly, regardless of the relative and absolute angles and positions of the object, viewpoint and the tool itself, we decided to call it the Gyrotool. Coupled with this Gyrotool is a display window that shows which shape is selected for manipulation, the current position and size of the shape, the type of manipulation, and the step size. This window also maintains a consistent position in the user's viewpoint.

Clicking on a shape within the VE (using the PC mouse) selects it for manipulation. When the shapes are created in the shape editor it is important to give them meaningful names as the selected shape's name appears in the display. To the right of the shape name is the step size. For early trials this was set to either 1mm or 10mm. Clicking on the current step size in the display toggles it. The available types of manipulation

(position, expand, shrink, colour, context) are shown on the right of the display in lower case. When one of these is selected it changes to uppercase. Further selections supersede earlier ones that revert to lower case. Once a shape and a type of manipulation have been selected, clicking on the Gyrotool buttons will affect the selected shape. Some types of manipulation do not have a directional element, for example, if 'colour' is selected clicking on a Gyrotool button, this cycles the shape colour through a previously defined palette of possible colours. When 'context' is selected the context of the object is made visible (making the context invisible increased the rendering speed, allowing more design productivity). If the manipulation type is set to one that affects size or position, clicking on the Gyrotool buttons moves, expands or shrinks the shape in the axis direction of the Gyrotool button pressed. Size and position changes are limited by the collision boundaries of neighbouring objects. Within a particular design some elements may be constrained in some way (for example the disc drive has a fixed size). In these cases appropriate constraints can be placed on those object attributes such that manipulated by the tool is limited or removed.

3.3 Case Study Two: The Virtual Factory

3.3.1 Introduction

As part of the MOVE (Manufacturing Operations in Virtual Environments) programme (Wilson et al., 1995), in 1994 the EPSRC funded a study with the title "Applications for Virtual Reality in UK Manufacturing Industry". A major part of this research was a 'Workshop Demonstration of VR Application'. This required a demonstration VE to be constructed that would illustrate the potential of VR as an interactive tool and as an integrating medium. This would allow the workshop attendees, representatives from 17 UK industrial companies, hands on involvement in VR applications including structured tutorials to allow participants to provide a formal evaluation of VEs and VR systems. The attendees were also asked to complete questionnaires on the use of VR in industry. This required the attendees to answer some questions about what they thought of the demonstration application, giving the author an opportunity to gain valuable feedback on the design of the VE.

The virtual factory was designed to show how VR might be used in several particular applications, and to demonstrate the added value obtained by integrating a number of applications in one VE. For example, changes made to a product design will have implications on the manufacturing process, operations and end-user considerations. In designing the virtual environment, emphasis was placed on interactive features and visual demonstration of cause-effect relationships between virtual objects and processes rather than high-definition visually impressive graphics. Of course, the representation of an entire manufacturing process to any great level of detail would require programming of an immensely complex virtual environment far beyond the resources available to this study. In addition, we wanted the virtual factory to show what potential user companies could build within reasonable time, capital and operational resources.

The virtual environment represents a manufacturing plant producing toy vehicles for 2-6 yr old children. The plastic body and roof components of the product are manufactured at this plant using an injection moulding process. The user has the facility to modify the product design in size and aesthetic qualities (figure 3.2), and is able to test the product's suitability for different users (e.g. children of different ages). The injection moulding

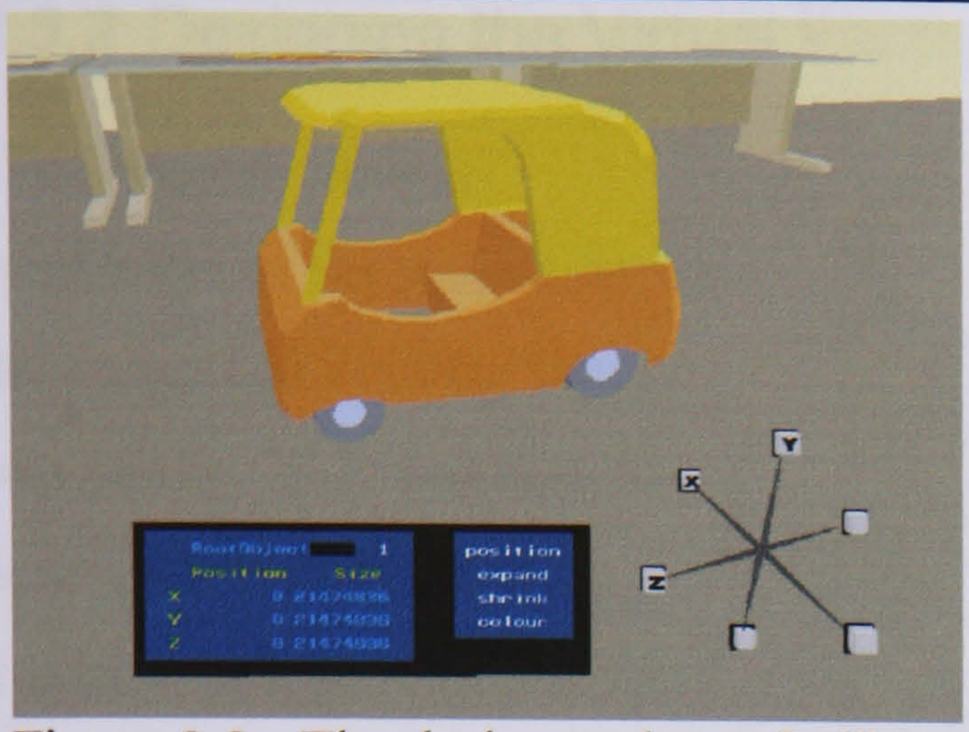


Figure 3.2. The design and test facility where the product, a child's car, can be modified.

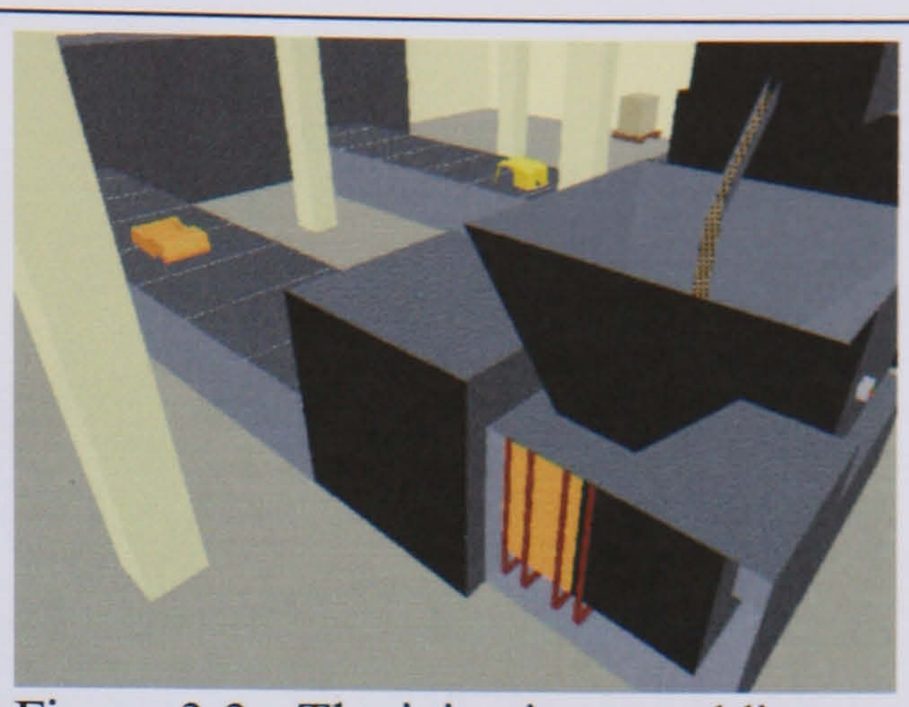


Figure 3.3. The injection moulding process, showing the components moving down the conveyer belt.

process is modelled and can be seen in operation allowing the components to be followed along the production line (figure 3.3).

The demonstration factory VE includes a design-manufacture-test facility to allow demonstration and examination of a number of attributes of virtual environments applicable to manufacturing, including:

- modelling in "virtual clay" - dimensioning, reforming and orienting, colouring.
- rapid prototyping through interactive design and test facilities.
- walkthroughs around a factory floor.
- rapid switching of viewpoints, at exocentric, egocentric and object-centred locations
- training, for operation or maintenance of equipment.
- visualisation of several stages in a manufacturing process.
- ergonomics assessment of "fit" between different user sizes and product dimensions.

The virtual factory was developed to highlight three of the main applications of interest to industrialists as identified through a National Survey and Follow-up surveys (Wilson et al., 1996). This allowed the demonstration to be divided into three modes: - factory walkthrough, visualisation of a manufacturing process and design modification. These modes formed three tutorials that each of the attendees had the opportunity to go through. At the end of each tutorial stage the attendees filled in response sheets indicating their impressions of the specific features demonstrated and their utility for industrial applications.

3.3.2 Designing the Virtual Factory

The time and funding constraints of the project limited the design and building time to two weeks. This was seen as potentially limiting the time available to apply some of the lessons learned previously. On the other hand it was hoped that the experience gained from earlier work would help to streamline the development process and produce a more complete VE than otherwise would have been possible in the time available. Much of the topology, many of the objects and some of the behaviours were imported from VEs previously built by VIRART for other purposes, such as the forklift truck (figure 3.4). Other elements imported into this VE included the gyrotool from the Virtual Prototyping project (see previous section) and a conveyer belt system that had been part of some fundamental research carried out in 1993 (for further details see Eastgate (1994)).

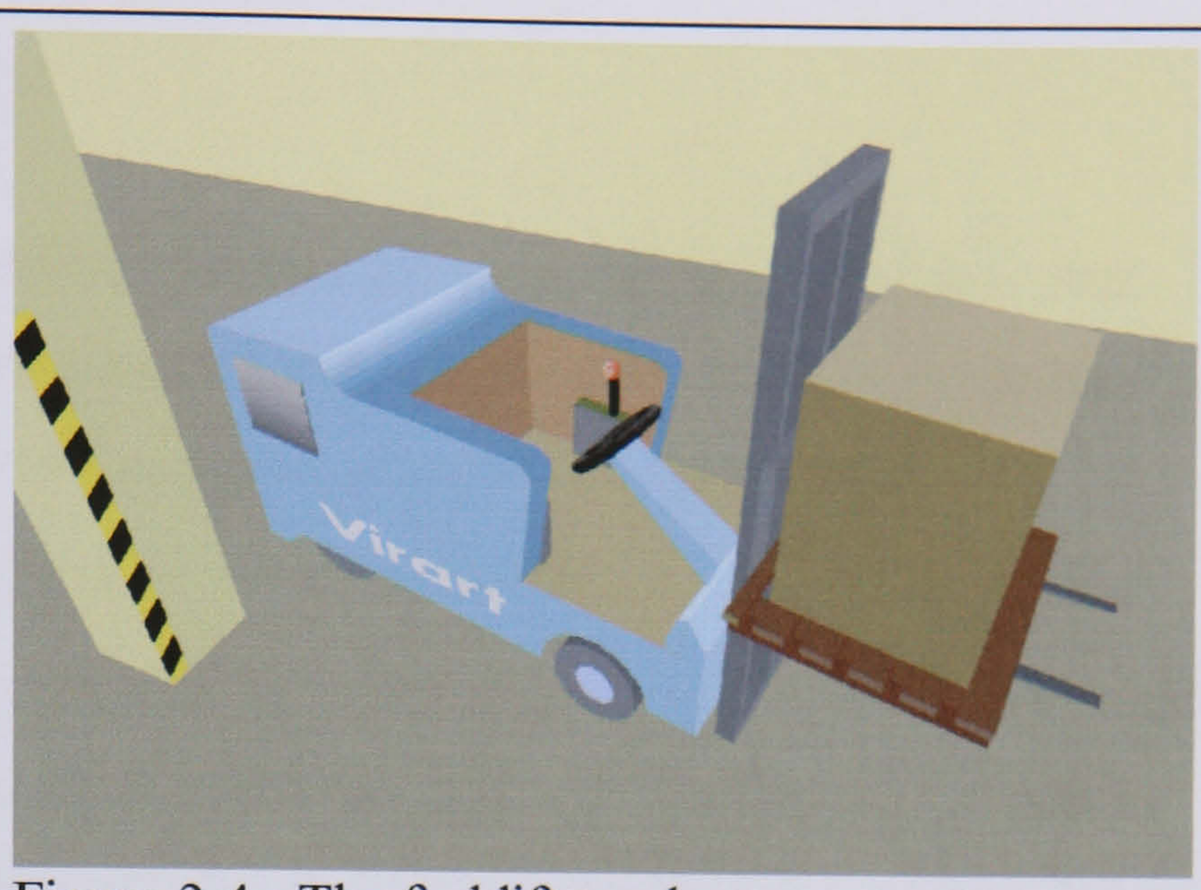


Figure 3.4. The forklift truck

From previous experience it was known that high levels of detail would reduce the rendering speed and take longer to model so most of the new equipment models were built using simple cuboid geometry. As much of the VE was representing generic manufacturing processes the simplicity of the model was not a problem so long as the principles of the processes were clear. The toy car was built using a higher level of detail, as this was the focus of the virtual design process. This did cause rendering speed problems later when many cars were 'produced' by the virtual production line, as each car produced increased the number of facets which needed to be rendered. During the workshop, the tutorials followed by the attendees were structured to avoid allowing time for large numbers of cars to be produced. The extent of the problem was only realised during less structured use of the VE later. Repeated duplication of objects will cause problems for any VR system eventually. When employing repeated duplication of objects some sort of management system should be incorporated such as, to reduce the

level of detail of the objects produced, or delete the oldest objects as new ones are created.

3.3.3 Evaluation

At the end of the workshop each attendee was asked to complete a questionnaire that asked for their general opinions of the potential take up of VR within their companies. Part of this questionnaire was aimed at finding out what they thought of the demonstration application. The results are summarised below.

- Usability of VR was measured in terms of how easy or difficult the respondents found it to perform specific operations. They differed widely in their ability to recognise where they were in the virtual environment, and to drive the forklift truck and lift the pallet. However, determining where to go in the virtual environment and moving around using walk view presented no difficulties to the respondents.
- In visualisation of the manufacturing process the respondents had no difficulty in identifying the most useful viewpoint to use but differed in their ability to select or move to the viewpoint required. Most of them found the use of "ghost mode" to fly into the machine and recognising where they were inside the machine very difficult.
- In design modification the respondents found all the operations, including selection of product components and design features, use of the gyrotool to change the product design, and viewing the consequences of changes in production and for the end-user, easy to perform.
- Users commented that the cues to aid navigation and interaction were insufficient and that the VE could have been improved to give a better feeling of presence by the use of sound.

From this it would appear that the users were satisfied with the visual and functional aspects of the various systems modelled in the VE, but had problems with the user interface, especially with respect to navigation. The variation in ability to use the navigation modes shows up the individual differences in users' aptitude for both the hardware navigation devices and the DOF provided by the software, with users tending to find the navigation modes with the most DOF (e.g. 'ghost mode') the most difficult to use.

When developing the VE it was decided to include some scenery surrounding the factory (figure 3.5). It was felt that this might perform several functions, especially when considering that the workshop attendees would largely be novice users of VR technology. It was hoped that this would;

- provide an introduction to the concept of VR and to this VE
- give a context to the factory making it seem more plausible and helping the user with orientation and navigation whilst outside the factory
- give the user a chance to get familiar with using the VR interface before encountering the more complex interactions inside the factory.

The responses to the questionnaire did not reveal what the users thought of the scenery provided but the lack of comment on it would seem to show that it was at least acceptable.

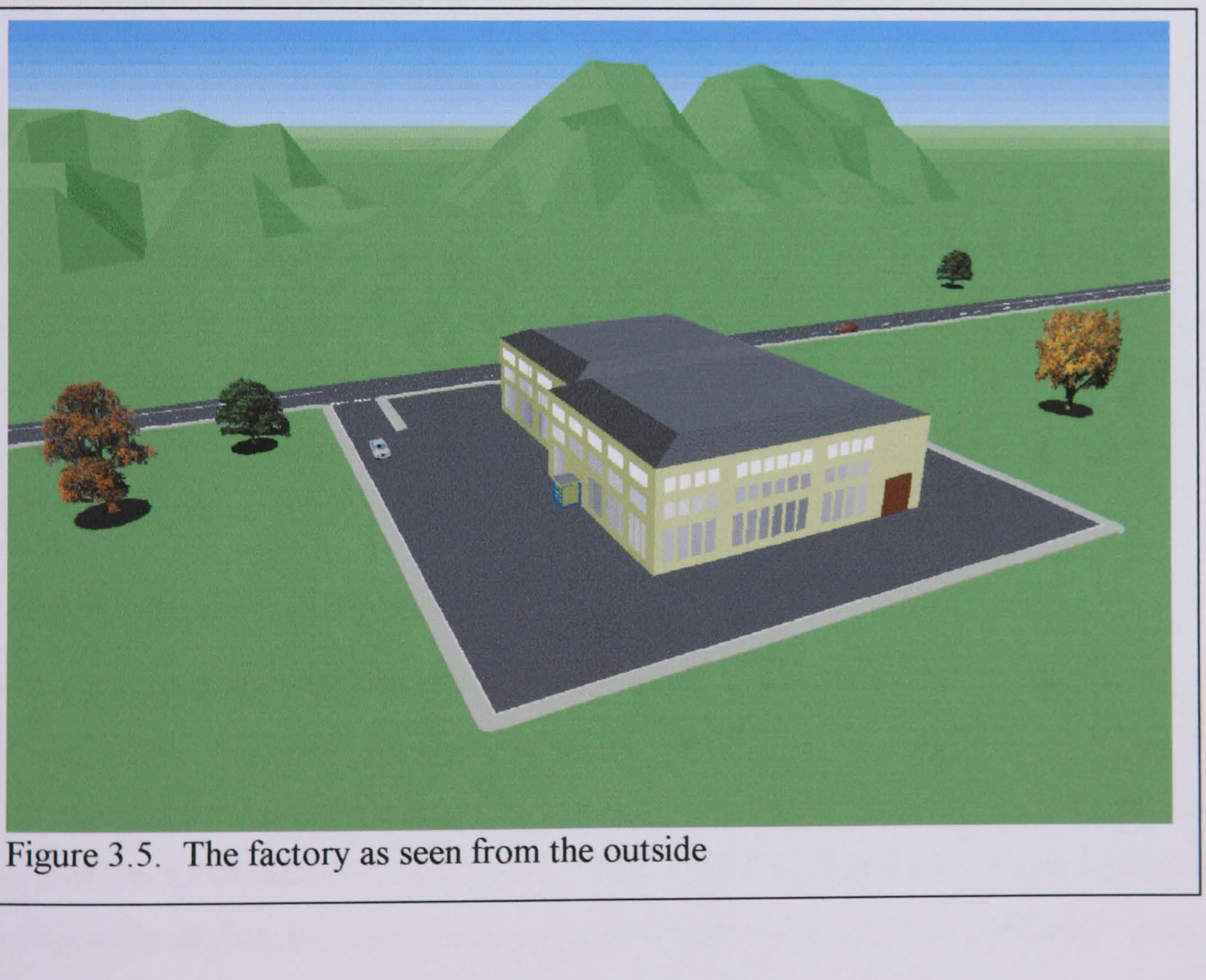


Figure 3.5. The factory as seen from the outside

3.4 Case Study Three: The virtual ATM

3.4.1 Introduction

At the end of 1995 VIRART were asked to build a Virtual ATM (automated teller machine or cash dispenser) that could be used to train bank employees to perform various replenishment tasks (Eastgate, D'Cruz, & Wilson, 1995; RM. Eastgate et al., 1997). It was clear from the outset however, that the underlying purpose of the project was to demonstrate to various groups within NCR the utility of VR technology. As explained by the 'VE experiences' model of the previous chapter, the aim was to motivate users so that they would explore the VE enough for it to achieve its objectives. It was therefore decided that the VE produced would need to be visually impressive in order to attract people's attention, and interest them enough to get them to consider VR's possibilities as a serious computer technology that could be applied to their business function. Bearing this in mind it was decided that the presentation of the VE would be at least as important as its ability to perform its training function. Building on previous experience, it was also thought that a VE containing just a single ATM would not give people who had no experience of VR much idea of its wider potential, and that the ATM should be modelled in context. It was therefore decided at an early stage to model not just a single ATM, but also the entire ATM demonstration suite at NCR's Dundee site (only one of the ATMs modelled would be given any functionality). By starting the experience outside the suite the user would have a short time to get used to the technology and the VE before encountering the complexity of the Virtual ATM replenishment tasks.

3.4.2 The Building Programme

Before any building work was started the author visited the NCR site in Dundee to see how an ATM is replenished, and to discuss the building of the Virtual ATM. An engineer was videoed carrying out the replenishment processes and around 70 photographs were taken of the ATM and the demonstration suite. Many of these photographs were taken orthogonally so that they could be scanned into a digital format and pasted onto surfaces in the VE for added realism. Detailed drawings and user manuals 'to be sent to us as soon as possible' were promised. Unfortunately these were never received and eventually when the work was getting seriously behind schedule a second visit was made to Dundee, armed with a tape measure, to get the dimensions required for building the model. A

training video was sent, which was useful, as it gave an insight into the existing training methods for this type of job, as well as showing someone performing the tasks.
Following user comments on the 'virtual factory' it had been hoped to be able to get some realistic sounds from the video but the soundtrack was not suitable. VIRART has a library of sound effects CDs and in the end some suitable sounds were taken from these. Ultimately the actual sounds of the real ATM could have been sampled and replayed by the VR system to correspond with the appropriate activity in the VE.

The VE was developed on a 90Mhz Pentium PC. Unfortunately, at the end of the project, the VE had to be demonstrated in Dundee using a 75Mhz Pentium PC without a sound card. Apart from the obvious absence of sound no serious drop in performance was noticed.

In this case study an environment that really exists was being modelled. In order to maintain an acceptable rendering speed it would be necessary to carefully decide which objects to model from those present in the room. The author's previous model of the

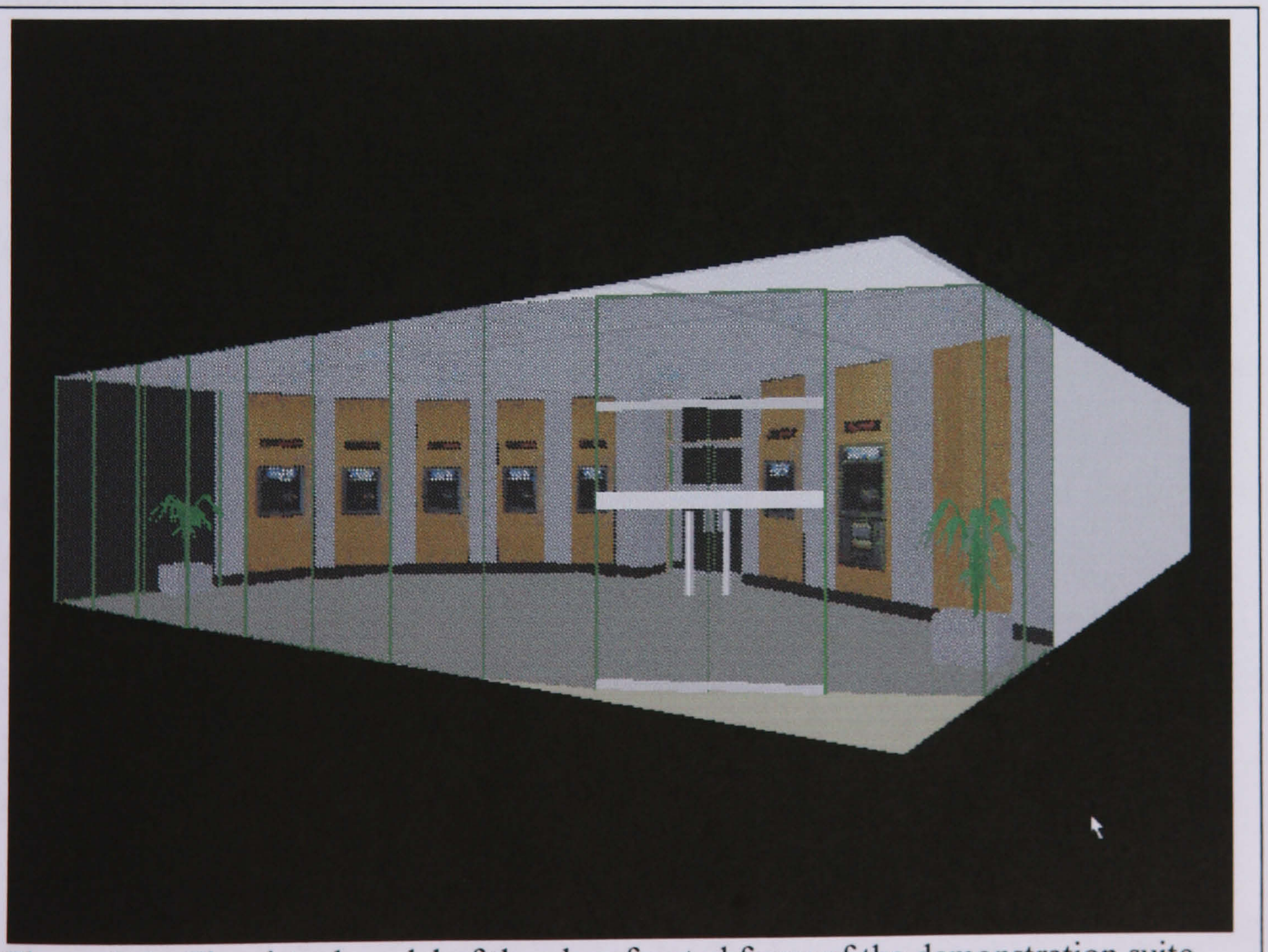


Figure 3.6. The virtual model of the glass fronted foyer of the demonstration suite

components of a VE, derived in chapter four, had named objects as one of the four building blocks. This category would now have to be further subdivided to enable the prioritisation of different object types. The categories of object types thought necessary for this VE (and maybe VEs in general) were; boundaries (walls, ceiling, doors), landmarks (doorways, pillars, prominent objects), obstacles, devices (tools etc), and features which aid recognition (pictures, other articles specific to that environment).

The demonstration suite consists of two rooms. The first is a glass-fronted foyer (see figure 3.6) with double doors leading into it. Six ATMs of various types are mounted in a wood panelled wall. Apart from some potted plants the rest of this area is empty. The shape of this room is irregular which creates some problems for the world builder. More double doors lead through to the second room (see figure 3.7). This room is more regular in shape but still features some odd angles. It is laid out like a bank with desks and counters and several free standing ATMs.

Figure 3.7. The second room with its free standing ATMs

An effective method of modelling rooms in Superscape's VRT is to create each one as a single shape with internal facets (a square room would be modelled as an inside out cube). This technique works fine as long as the shape of the room does not incorporate any intrusions (e.g. a chimney breast) as from some angles these will need to render in front of the room's contents. If a room does have intrusions it can either be broken up into two shapes, or the room can be built without the intrusions, which can then be added later as if they are objects in the room. Both of these techniques have been used when building the irregular shaped rooms of the demonstration suite.

The demonstration suite was built to place the Virtual ATM in context and to give novice users a chance to explore a VE giving them experience of the technology. As such it didn't have to be modelled accurately but it did have to look right and be modelled

correctly for navigation (landmarks and obstacles). Detail that was omitted included; the foyer ceiling (a false skylight made in an intricate pattern of wood and glass) as users are less likely to look up and rarely use the ceiling to recognise or navigate through an environment; some of the counters in the second room as these would have been time consuming to model and did not qualify as important boundaries, landmarks, or obstacles; signs not relevant to the purpose of the VE; and office equipment (phones etc) also not relevant.

All of the walls, floors and ceilings of both rooms were modelled so as to form boundaries through which the user could not pass except via a doorway. The modelling was done entirely from photographs and so is not accurately dimensioned. It does look right however and no users who had visited the actual rooms criticised the appearance of the virtual rooms. Textures were added to some of the wall facets to make the rooms recognisable. These include wall mounted ATMs, glass panels, pictures and display units. These textures were created from photographs taken in the actual rooms.

Glass doors were modelled in the doorways. Like the real doors these open both ways and were programmed to open away from the user as they approach them in the VE. Other objects modelled included chairs, plant tubs, a pillar, and some non-functional ATMs as well as the functional ATM. Where applicable, textures such as wood grain were added to these objects. These objects were all given collision boundaries such that the user cannot pass through, but must move round them.

Figure 3.8. The ATM with its three access doors open

The methods used in building the ATM itself were dictated by the specifics of Superscape's VRT especially with respect to object boundaries and hierarchical structure. When closed up, the free standing ATM is relatively small and self-contained. There are three doors that open to give access to the modules that need replenishing (see figure 3.8). Two of these doors open sideways (one each way) and give

access to the lower part of the ATM. The user console however lifts up to give access to the upper part of the ATM. As these doors are opened they increase the amount of space taken up by the ATM as a whole. In order that the ATM continues to render correctly it is important that its logical boundary increases in size to include all parts of the ATM in each configuration. This problem is repeated as the modules are racked out for replenishment. As they are pulled out they occupy positions previously occupied by the doors. Superscape's VRT does not automatically cater for these factors so extensive programming (SCL) had to be done to change the logical boundaries of objects as required to maintain the arrangement of the hierarchical structure for correct rendering. When a spent component is removed from the ATM its position in the hierarchy has to change, as it is no longer a part of the ATM (and the new component has to become part of the ATM). This again requires some complex programming.

The real ATM is opened with a key. The keyhole is positioned out of sight under the bottom front edge of the console. It is not the purpose of this VE to train people in the skills of using keys, rather it is to inform them of the need to use a key, so the interaction required to unlock the ATM has been simplified to a mouse click on the keyhole. This in itself gives the VE developer some problems. When a person unlocks a lock that is one metre (approx.) above the ground, and pointing down, do they do it by touch or do they bend down so they can see the lock? In VR it is normal to interact with devices that you can see, so the bending down solution would seem appropriate, but how do you simulate bending down in desktop VR? Normally interaction has self-explanatory visual feedback confirming the success or failure of the action (the door opens, the light comes on). With no tactile feedback mechanism available and bearing in mind the size of a keyhole, how can you give the user a clear indication that the unit is now unlocked? As is true for much work in VR, it was necessary to settle for an adequate rather than a perfect solution. The possibility of performing the task by touch rather than by sight had to be ignored. The bending action is achieved by selecting a new viewpoint (by pressing a button on the spacemouse) that is low enough to see the keyhole. Feedback is provided in the form of a clicking sound that approximates to the sound made when unlocking the real ATM.

To improve the rendering speed all the internal workings of the ATM were programmed to be invisible unless the ATM doors are opened. Once the console is lifted up access can be gained to a two-position switch that switches between normal and supervisor mode. In

normal mode the ATM screen is as a customer would find it. When supervisor mode is selected a series of diagnostics screens inform the user of the 'health' of the ATM. They are updated to show the current 'health' of the ATM as the replenishment tasks are completed. All of these screens are textures created from photographs taken of the actual ATM screens.

The task modelled was replenishment of a currency cassette (refilling one of the cash boxes with notes) that includes the following sequence of actions: -

- Unlock and lift up the ATM console
- Switch to supervisor mode and check the health of the machine
- Open the lower door
- Unlock and open the safe
- Pull out the currency cassette and place it on the table
- Open the cassette lid
- Refill cassette
- Close cassette lid
- Slot cassette back into currency dispenser module
- Check health of machine again
- If healthy switch back to normal mode and close all ATM doors

This sequence of actions has to be performed in the correct order if the task is to be completed successfully. The effects of all of these actions (whether correctly applied or not) had to be modelled to give the participant the feedback required to make this a worthwhile training package.

As a visual cue to the user, on the real ATM all the points of interaction (switches, levers etc) that are required during normal maintenance and replenishment procedures are colour coded bright green. This is also helpful on the Virtual ATM as discerning which objects afford interaction is generally more difficult in a VE than in real life. If a green colour coded object does not afford interaction at the time when an interaction is attempted (e.g. because a door is locked), a message box appears stating "this action is not possible at this time", thus confirming that the object normally, or sometimes affords interaction.

Some objects' behaviours incorporate significant visual feedback. Where this is not the case another form of feedback needs to be added, possibly audio or textual. When these issues were discussed with NCR they suggested that the Virtual ATM could be used to help design ATMs for usability. They hypothesised that if an interaction was too difficult to model in VR it was probably too complicated in reality, and an alternative method should be developed. This could be a suitable area for further research.

During the process of replenishing the currency dispenser the currency cassette has to be removed from the ATM. In VR terms this means that the currency cassette must cease to be a child of the ATM and become a 'movable' object. Movable objects can move between areas (e.g. rooms) in a VE without being children of these areas. They will render correctly provided that their logical boundary is entirely inside or outside all other object's logical boundaries. If however a movable object is part inside and part outside the logical boundary of an object or area it may render incorrectly. This means that it is inevitable that during the process of removing and replacing the currency cassette there will be some rendering problems. The only ways of preventing this are: -

1. To increase the step size of an object's movement such that it jumps from inside to outside a logical boundary. This results in unrealistic movement.
2. To restrict the possible viewpoints to ones from which the rendering problem cannot be seen. This is not true VR and negates some of the benefits of using a VR system.

Neither of these solutions were implemented, as the existing problem was perceived to be so small and temporary as to be less than the side effects of the possible solutions.

Having removed the currency dispenser from the ATM we then have the problem of how to proceed. In the real world there is an almost infinite number of ways in which a person could put notes into a currency cassette, starting with where the cassette is placed during the procedure. Because of the inadequacy of the input devices it was decided to largely automate the procedure leaving the user to instigate actions by interacting with the correct devices (switches, levers etc), triggering animated sequences depicting the stages of the procedure. Thus the user learns the specific principles, but does not have to perform the more mundane components, of the task.

Modelling flexible objects in VR always causes problems. Modelling the behaviour of hundreds of currency notes was considered to be beyond the scope of this project. Instead

the notes are treated as a block that is animated where necessary. An important part of the procedure is to fan the notes before they are inserted into the cassette. Again due to the inadequacy of the input devices the user is just reminded (via a text prompt) that this should be done, but is not required to do it.

The process of replenishing the currency dispenser as modelled, is reversible and can be repeated ad infinitum. A section of the program controlling the process checks whether the cassette is in and whether it is full, and the supervisor display is updated accordingly.

Modelling the receipt printer again involved the problem of how to model flexible objects, in this case the paper on which the receipts are printed. This caused a problem because: -

1. The behaviour of a flexible object is a complex function of many different variables and so would use up a lot of processor time to calculate.
2. The resulting shape of the object may be difficult to model.

VR developers get round this by modelling the object in a few typical configurations and then animating between them as appropriate. The result is a representation rather than a simulation of the object and its behaviour.

As a result of this modelling restriction and the interaction restrictions discussed earlier, the training of this replenishment procedure is strictly limited to giving the user the correct conceptual model of the system, concentrating on the points of interaction, the sequence of actions, and the required results. Anyone who has replenished a printer will know that there are many pitfalls resulting in torn or jammed paper. These could not be modelled in a project of this scale, but the more common ones could be represented as part of a future extension to the project. As before all interactions are instigated by clicking on objects with the mouse, including complex manipulations such as threading paper into the mechanism. As the user completes each correct interaction, the mechanism and components are animated to show the salient points of the procedure.

The printer unit is an irregularly shaped and complex assembly. To model it geometrically accurately would have been a month's work on its own. Instead textures have been used which incorporate invisible areas to make up the complex shapes. The technique used is as follows;

1. A photograph is taken orthogonally of the assembly.
2. The photograph is scanned into a digital format
3. An art package is used to colour all of the background in the photo with the first colour in the palette, as this colour becomes invisible once the image is imported into Superscape's VRT.
4. This image, however complex, can then be pasted onto a single rectangular facet in the VE. This gives a 2D representation.
5. By layering several of these images the 3D impression of a dense and complex mechanism can be achieved. As these images have been taken from photos of the actual assembly the result can be accurate and afford good recognition.

There is one drawback of this technique. Normally when using Superscape's VRT an object which affords interaction and is visible, can be activated by mouse clicking on it. However interaction is not possible by mouse clicking through an invisible area of a textured facet. The user has no means of telling whether an area is or is not an invisible area of a textured facet. As a result they can become confused by this inconsistency in the user interface.

3.4.3 Conclusions

A successful aspect of this project was the use of the 'VE experience' model from the previous chapter to enable the recognition of the need to motivate the user in order to satisfy the client's objectives for the VE. This project also showed up the need for further sub categorisation of the components of a VE in order to help make decisions on whether to include or omit objects during the VE building stage.

3.5 Case Study Four: Applying virtual environment technology to a training application

3.5.1 Introduction

Based upon previous work covered in earlier case studies, particularly work for NCR. and to a lesser extent educational projects, it was realised that further research was needed to establish the role that VR technology could play in training. To do this it was decided to set up a laboratory based experiment in which a direct comparison could be made between a VE Training (VET) application and traditional methods (D'Cruz, 1999; D'Cruz, Eastgate, & Wilson, 1997; Eastgate, Nichols, & D'Cruz, 1997). As obsolete computers were available, a computer maintenance task was chosen where training using a VET could be compared with training using a video, and a control group who had no training. Each group could then have their performance measured when performing the task in the real world.

A number of experts were consulted about the possible tasks involved in computer replenishment and maintenance. From these, the task of replacing a network card in a computer was chosen. It was decided that VET would be appropriate because interaction with the equipment in the real world is the most effective way of learning a task, but it is not always possible to have a spare computer to practise on. Also this particular task, if done incorrectly, can be expensive as there are a number of delicate components.

The initial specification was translated into a VET application after consideration of a number of issues. The first was how the VE would provide instructions to the user. This was tackled by using a hybrid display (a combination of both 3D and 2D elements on screen at the same time) with the screen laid out in three parts as shown in Figure 3.9. The main part of the screen is a window into the VE (3D). This is the only part of the screen that the user interacts with. The other (2D) parts are to provide information. Below the VE window are textual prompts that change as the user completes each required step, and to the left are 'picons', which will be discussed in the next section.

3.5.2 Development of the VET

Firstly there was the issue of how to present the relevant information in the VE to the user (software user interface). In general, this is determined by the specifics of the hardware user interface and the requirements of the tasks as defined in the specification. After much discussion it was decided to divide the screen into three parts as shown in Figure 3.9. The main part of the screen was a window into the VE. This was the only part of the screen the user interacted with. The other parts were to provide information. Below this window were textual prompts to provide the instructions as identified through the task analysis. These automatically changed as the trainee completed each step. If an error were made, a textual prompt would appear to inform the trainee.

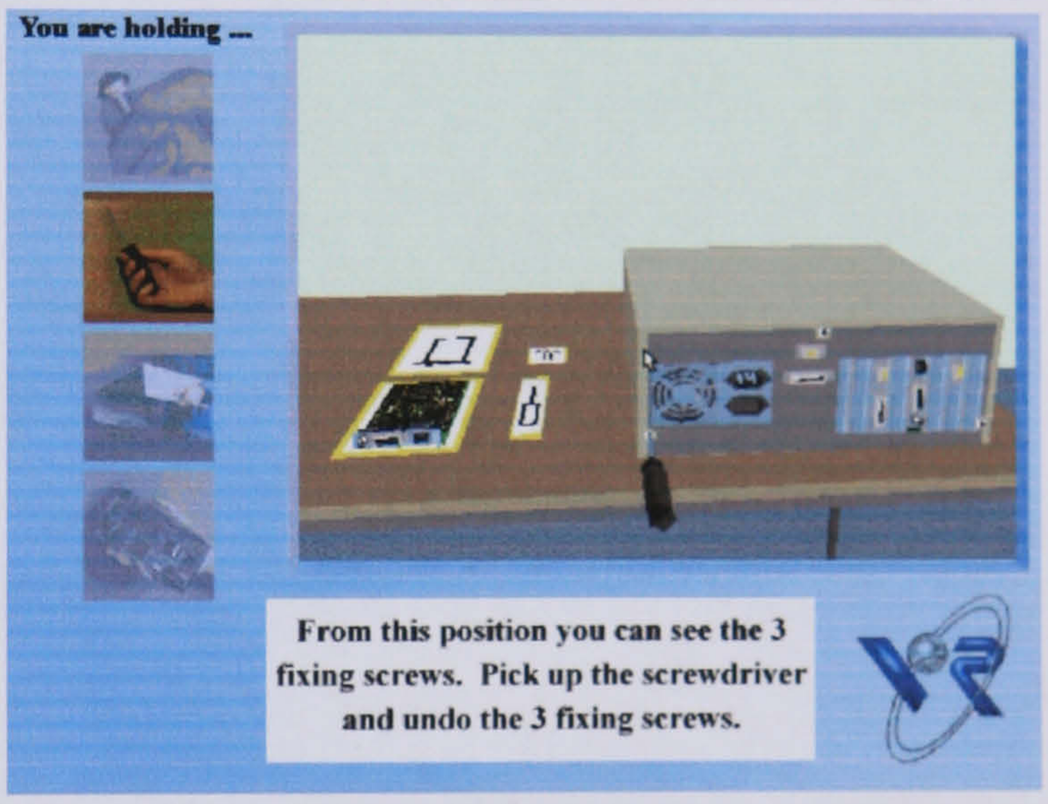


Figure 3.9. Layout of screen

Figure 3.10. 'Holding' the screwdriver and three screws

A second issue was how to represent the user in the VE. As in the ATM case study, due to the constraints of the desktop system the user had no clear representation (e.g. no body or hand) in the VE. However, the task required objects to be "picked up", "placed down", "lifted off", "taken out" etc and at times it was necessary to place certain items down on a table before others were picked up. Problems were encountered with representing the psychological processes (psychological fidelity) correctly, as well as the physical processes (physical fidelity) (Goldstein, 1993). For instance, in the real world, it is often necessary to have both hands free in order to perform an action like 'lifting off the outer casing of the computer' and it is obvious when your hands are not free because you feel the 'presence' of a screwdriver or screws without

seeing them. However in the VE, where there are no virtual hands to be occupied or 'feel' anything, it becomes necessary to represent these physical circumstances in an alternative obvious way. To have not considered this problem would be to train users incorrectly and may have resulted in 'negative' transfer to the real world of the task. that is the training may actually hinder performance of the real task (Goldstein, 1993). The picking and placing procedure was split into four parts; picking something up, holding it, performing an action with it, putting it down. The metaphor for picking something up in a desktop VE is usually the simplest, i.e. clicking on it with the mouse (or other activation device). This metaphor was adopted here. In the virtual ATM VE (section 3.4), an assumption was made by the developer as to what the user would want to do with the object picked up (e.g. the roll of paper), and the object would then perform that activity by itself. This resulted in objects within the VE floating through the air in a ghostly manner. In this VE it was decided that objects shouldn't float through the air, but more importantly for this to be an effective training tool, that the users should decide themselves what to do with the objects they picked up. Therefore picons (picture icons) were used as a metaphor to provide feedback to the user on what is present in their 'virtual' hand. Four picons were located to the left of the screen, one for each of the objects (or groups of objects) that could be picked up by the user. Initially all of these picons have a dull or washed out appearance. This indicates that while it is possible for the user to hold that object, they are not holding it at the current time. If the user does pick up an object (e.g. by clicking on it), then the picon representing that object gets its full colour and contrast, indicating that the user is holding that object (figure 3.10). At the same time the corresponding object in the VE window disappears. Where more than one object of the same type can be picked up (e.g. screws), the quantity currently held is shown by a number superimposed over the picon. When objects are put down the picon for that object becomes dull again. For performing an action with an object being held, the metaphor used was to click on the subject of that action. For example, when holding the screwdriver, a screw could be removed by clicking on it.

The final metaphor required was that for placing an object down. In the real world one can let go of anything being held at any time and it will fall onto the surface below, this is the opposite of grasping the object. Complex algorithms could be written to simulate this in VR but it was decided that this would not enhance the training process in this case. In a VE the object has been picked up by clicking on it with the left mouse button, there is

no obvious opposite of this action. To get round this each functional object was given a 'home'. These homes were marked by a simple picture of the relevant object (some of these homes can be seen to the left of the PC system unit in figure 3.9). Once an object had been picked up it could be put down again by clicking on the home. These homes were also included in the real world experiment so the training accurately matched the real world task. The combination of the use of 'homes' and 'picons' gave the user a consistent system of cues and feedback for, and total control over, all their picking and placing tasks.

Then, to give the user more feedback and add context to the experience, realistic sounds sampled from the real task were added to the VE. This was so that, when the users were performing the real task, they would be reassured on hearing sounds they remember from the VET and may react with caution on hearing an unfamiliar sound (e.g. a component being strained or broken).

This task includes one particularly difficult action. The new card has to be eased carefully but firmly into its socket. This requires quite a lot of force, but it can be made easier by pushing first on one end of the card and then on the other. This is represented in the VE as follows; when the user is holding the new card and clicks on the socket to insert it, the card first moves into position, then rocks slowly in its socket, before dropping home. In the instruction box it is emphasised how this activity requires both hands. Accompanying sounds serve to emphasise the difficulty in performing this action.

Another issue was how to allow the user to view the task in a way that related to the real situation. The input devices specifically available for desktop VR that best suited the interactions required were chosen. The keyboard was used to change viewpoints, and the spacemouse for movement and navigation. It was considered that two viewpoints - an 'average person standing' and 'average person bending' - would be adequate for completing the task as these were the two viewpoints generally used when completing the real task. The spacemouse was fixed to an average person's viewpoint with all objects given collision boundaries, as opposed to allowing complete freedom (that is, the ability to see the situation from any view e.g. bird's eye/worm's eye or ghost views). This was to counteract usability problems experienced by initial users during the review sessions, that is the spacemouse was found to be difficult to control and the many viewpoints

tended to add further problems. It was easy for the user to become 'lost' in the environment.

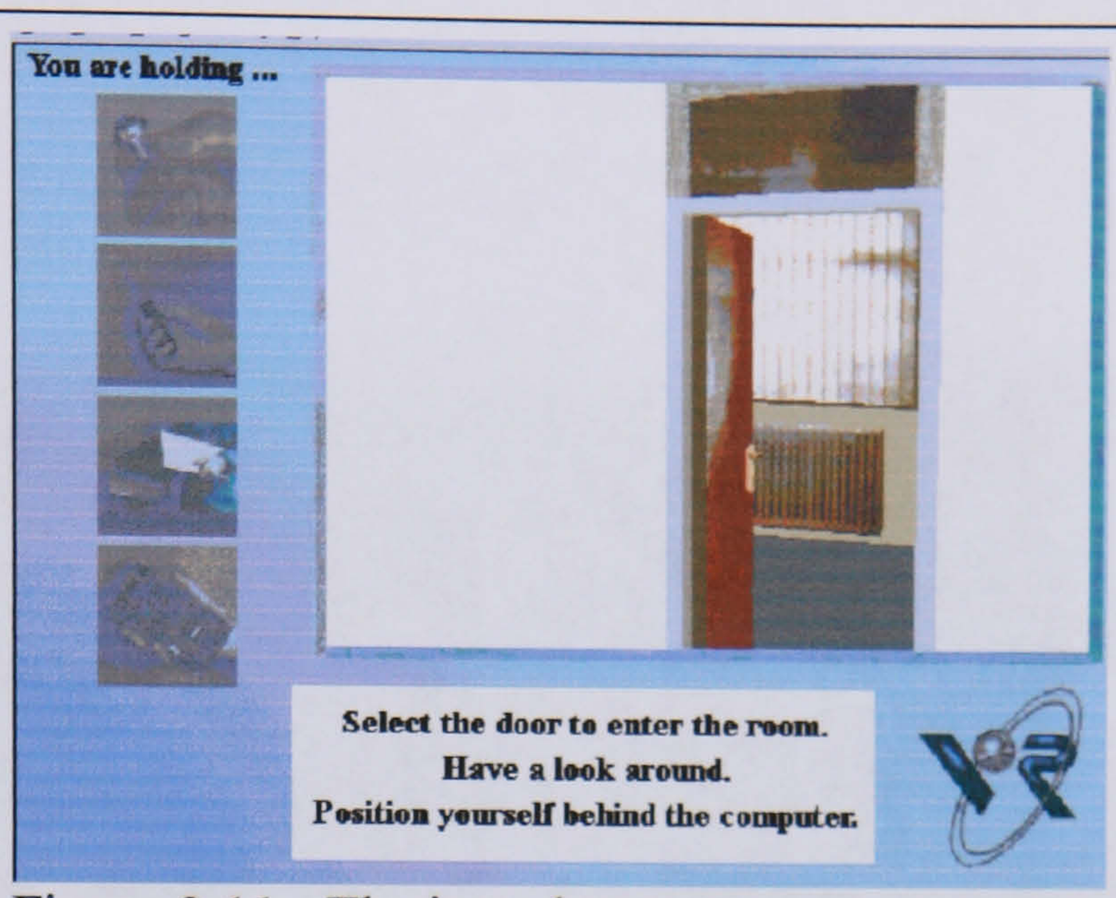


Figure 3.11. The introductory screen

In order to allow users to familiarise themselves with these functions, the task was modelled in a 'virtual' experiment room that was modelled on the room where the real world task would take place. The first textual prompt instructs the user to, "*Select the door to enter the room. Have a look around. Position yourself behind the computer*" as shown in Figure 3.11. The user is therefore required to use the mouse to open the door, the spacemouse to navigate through the room and the keyboard to select the 'bending' viewpoint to view the objects on the table.

A further design stage in the development was deciding which non-essential objects to include in the virtual experiment room. The actual room used for assessing user performance of the task is a typical office environment, which is cluttered, and contains many items not relevant to the task. The number of objects in the environment had to be reduced to maintain an adequate rendering speed and to decrease development time. The dilemma was in deciding which objects to include and which to omit. These decisions have to be made on both the macro and the micro levels, for example, when modelling a virtual room the resolution for the carpet texture may need to be decided, additionally omitting entire items of furniture may be deemed appropriate. What is more usual is that smaller details such as electric mains sockets are left out. As well as being guided by previous experience in building VEs (see section 3.4.2), human performance theory was applied in the form of Rasmusson's (1986) well known SRK model of skill-, rule- and knowledge-based behaviour to see if it could help choose which objects to include whilst reducing distractions, improving user orientation and reducing potential user errors in the VE. This approach was written up in Eastgate (1997).

3.5.3 Evaluation

An expert walkthrough appraisal of the initial model was carried out first to check that the VE was an accurate representation of the real world task. Modifications were made accordingly. Then a series of experiments was carried out to evaluate the effectiveness of the application for training and the good and bad features of the VET application.

Thirty subjects were randomly divided into three groups. The control group received no training before the task. The second group were trained by a video of an expert demonstrating the procedure. The third group were trained by the VET application. All the subjects were then required to carry out the task in the real world as quickly and as accurately as possible while being timed and videoed (figure 3.12). After this, all the subjects had to complete post-questionnaires aimed at discovering what the subjects thought were the good and bad aspects of the training, given the VR system and the VE.

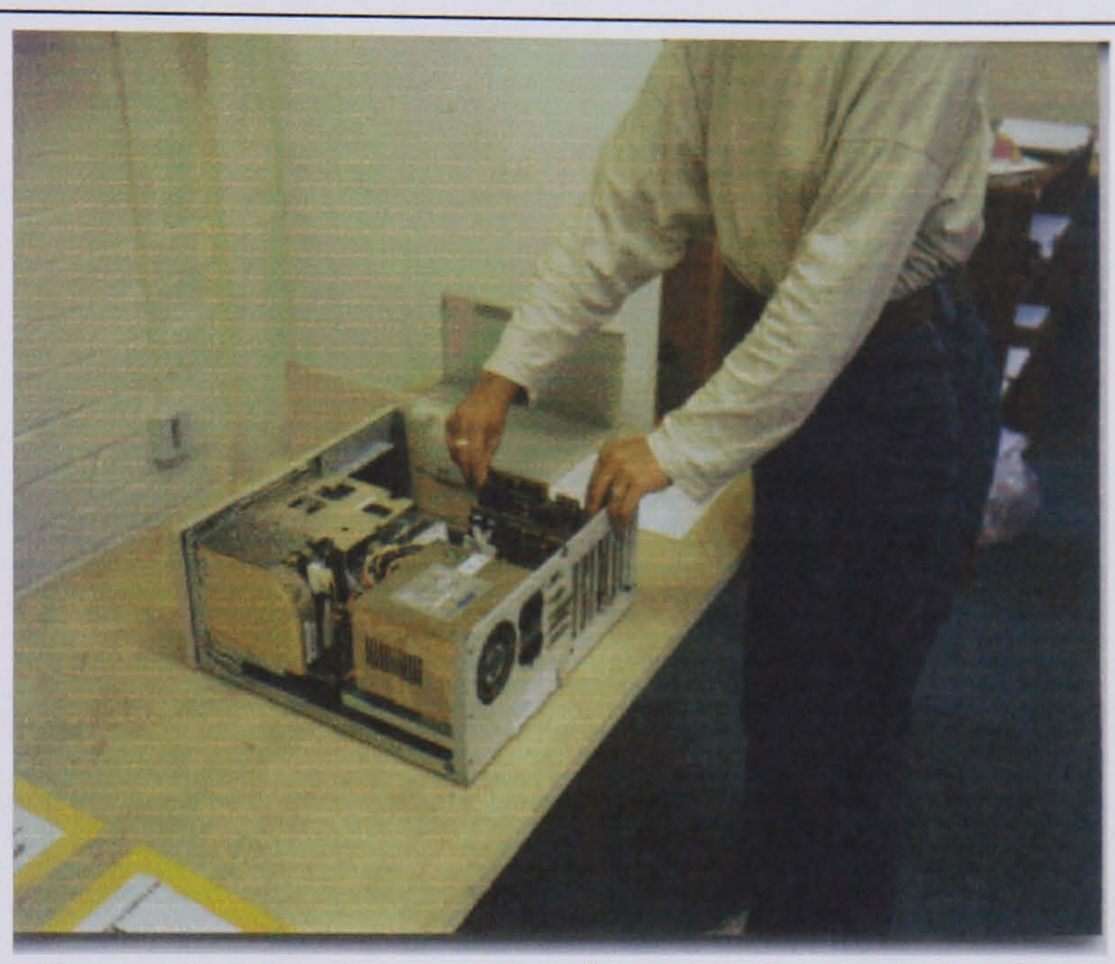


Figure 3.12. A user carries out the task in the 'real world'

The subjects of the video group found the method easy and effective. However only three thought it was 'interesting,' four found it 'boring' and four did not even venture an opinion on this description. All the subjects in the VET group agreed that the method was interactive, self-pacing, interesting and effective. Generally in most cases the VET group tended to enjoy the training method more than the video group. .

Evaluation criteria were developed to evaluate the features of the VET application. These were based on human computer interaction requirements and requirements specific to VE interaction. The components of the application were categorised and the subjects were asked to rate them individually then asked to give an overall 'satisfaction' rating. In this way it could be seen if any part of the VET was unsatisfactory and what particular feature

of that part was influencing this opinion. The features evaluated included the layout of the screen, visual appearance of the VE, use of sound, use of textual prompts, use of pictorial prompts, movement around the VE, use of different viewpoints. interaction with objects, behaviour of objects and sense of involvement and presence.

In general, all the subjects found using the VE straightforward. After going on to perform the real task they generally agreed that using the VET had adequately prepared them for what was involved. However some users commented that the VE hadn't prepared them for the physical effort involved in some aspects of the task. Overall the VET was perceived as effective as, but more motivating than the video in training the task and the subjects were satisfied with its features. However, the subjects had divided opinions about the features of the VR system, in particular, about the use of the spacemouse. This had implications for how much control of movement through the VE the subjects felt they had. Four subjects found the spacemouse difficult to use and for one particular subject this difficulty strongly affected enjoyment. Therefore either the initial training given to the subjects was not sufficient, or a different input device was required to better enable novice users to navigate round a VE using a desktop VR system.

3.6 Discussion

The new issues raised in case studies two to four (table 3.2) can be categorised for discussion into three inter-related areas; navigation, interaction and reducing the number of objects. The rest of this discussion will deal with each of these areas in turn.

	The virtual factory	**The virtual ATM**	**Changing a network card**
NAVIGATION			
Context	Surrounding scenery	Demonstration suite	Room
Input devices	Use of a spaceball		Use of a spacemouse
INTERACTION			
Cues and feedback	The use of sound. No cues on machine cover Detailed instructions	Only real world cues	Extensive user interface Detailed instructions
Object manipulation	Forklift use problems	Metaphors replace self embodiment	Improved metaphors
User interface			Picons, context sensitive instructions, hybrid display
REDUCING NUMBER OF OBJECTS			
Development time		Reducing object count	Reducing object count
Focus vs. distraction		Choice of objects to include	Choice of objects to include
Realism		Heavy use of textures	Heavy use of textures
Rendering speed	Object duplication	Reducing object count	

Table 3.2. The VE development issues raised by each case study

3.6.1 Navigation

Waller et al (1998) divide navigation up into interface fidelity and environmental fidelity. However they were concerned with using VEs as a direct replacement for a real environment without using the capabilities of VR technology to enhance environments to improve usability (it is not this author's view that the only aim of VR is to somehow, veridically, recreate a reality). The development of a VE can be an attempt to accurately model an actual or possible reality (as was partly done in the 'virtual ATM' case study), whereby within the limits of the system an attempt is made to simulate an existing environment, or one that could exist, with as much fidelity as possible. Alternatively, the

shortcomings of the system can be recognised and compensated for by deliberately distorting the representation of the environment such that it communicates relevant information more effectively (as was done in the 'changing a network card' case study). The less sophisticated the VR system and the less the concept to be communicated lends itself to VE representation, the more pronounced these distortions may have to be. In the extreme case this can involve the addition of totally unrealistic elements in the VE. These elements could include text prompts that appear in front of the user, audio prompts, or video clips. Earlier work by Darken and Sibert (1996) compared different types of VE with varying levels of superimposed grids and maps to find out what effect this had on way-finding behaviour (purposeful, oriented movement during navigation). As a development of the 'Interfaces in VR System Use' model (see fig 2.1), which splits the VR-user interface into the physical and cognitive, navigation can be viewed as having a hierarchy of components (figure 3.13). At the top level it can be split into the interface and the environmental components, where the interface includes all aspects of the user's control over movement in the VE and the way in which the content of the VE is relayed back to the user. The environmental component is the VE itself.

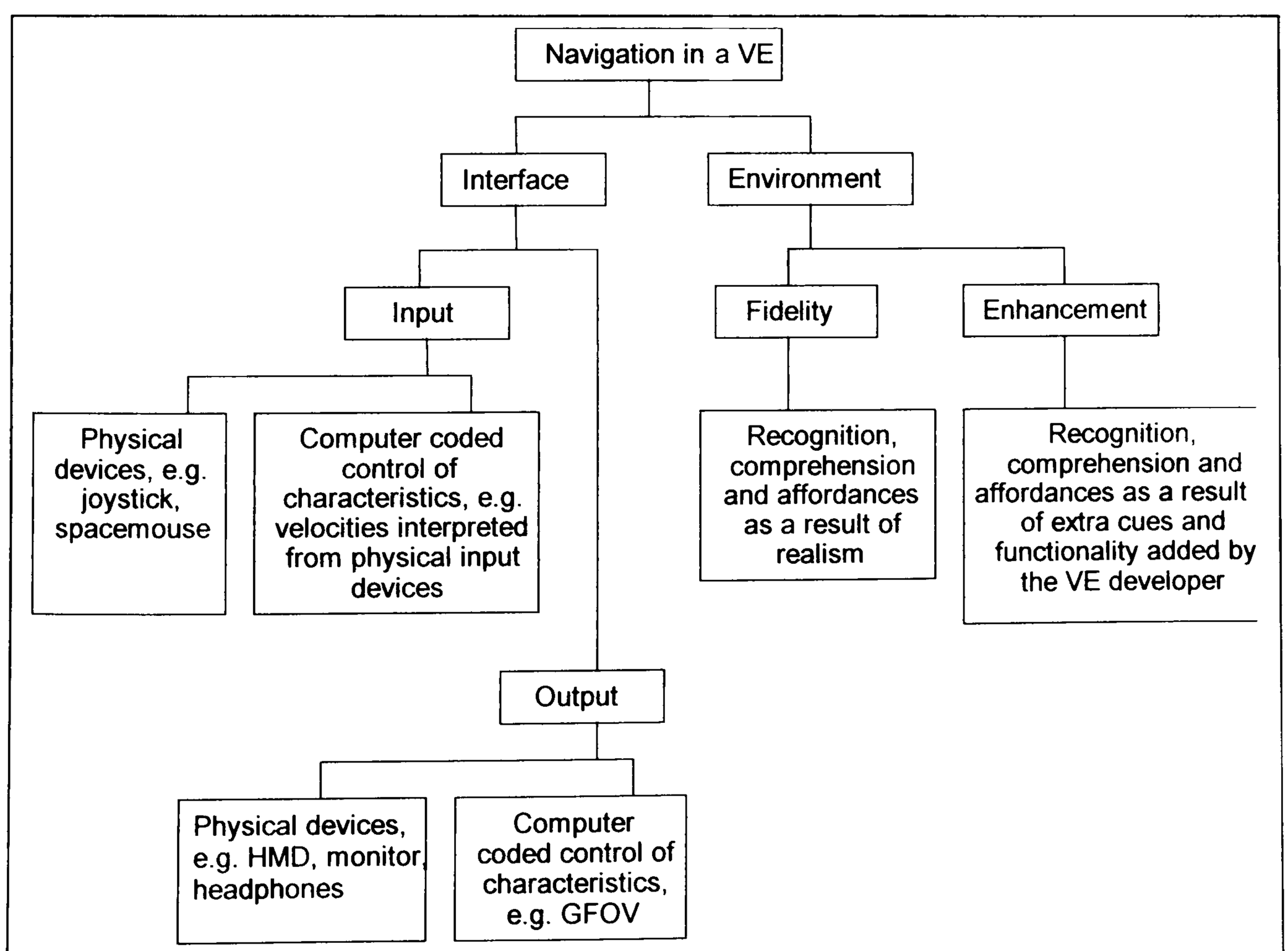


Figure 3.13. The hierarchical structure of the components of navigation

The interface element can be further split into the input and output systems used. The input systems include the physical input devices and the computer coding that converts these inputs into virtual movement. This virtual movement is made up of the linear and angular velocities and rates of acceleration interpreted from the input, the number of degrees of freedom offered by the input system (see section 4.3.4), and any constraints on the user's movement such as the effects of gravity. It is widely recognised that the physical input devices available have significant limitations, the more versatile are difficult to use, the ones that are easiest to use afford limited freedom of movement. The output systems include the characteristics of the physical display devices used such as the resolution and the physical field of view, and the characteristics of the software driving the displays including the geometric field of view (Waller, 1999) and the angle and position of the viewpoints available. To these output systems can be added the audio and haptic feedback devices where these are used to aid navigation. All of these interface components have levels of fidelity and usability amongst their characteristics. Here fidelity refers to how faithfully the interface emulates real world interactions and how accurately the interface transfers information and interactions from the user to the VE and vice versa. It is the general low level of the fidelity of the interface systems (especially the hardware) that leads to the requirement for enhancing the usability of the VR systems in general and VEs in particular. As Waller states in the context of VET "perfect fidelity would yield transfer equivalent to real world training because the two environments would be indistinguishable" (Waller et al., 1998), although it should be remembered that totally realistic training is not necessarily the best training.

The environmental element of navigation can be split up into environmental fidelity and environmental enhancements. In this case environmental fidelity refers to the realism of the 3D model(s) as built by the VE developer. The fidelity of the VE will determine whether the user can recognise objects within the VE and comprehend the distances between and relative positions of the objects within the field of view. This recognition and comprehension will allow the user to take advantage of the affordances offered by the VE, such as walking down a path or moving through a doorway. As mentioned in the previous paragraph, the lack of interface fidelity leads to the requirement for enhancements in the VE to compensate. Many of these enhancements are based on navigational aids found in the real world, but in a VE the need for them will be greater

and their implementation can usually be more sophisticated. These enhancements could be in the form of arrows or paths showing the user where to go, a map and/or a compass overlaid or inset into the viewpoint, or the ability to view the VE from above to get an overall view of its layout. Other enhancements could change the nature of the affordances themselves, for example, an enhancement suggested by Neale et al (2000) is that increasing the size of corridors and doorways can make navigation simpler, especially for users with (real world) disabilities. Darken and Sibert (1998) recommend that VEs should be given an explicit structure, dividing the world into small distinct parts, to allow the user to mentally organise the environment. Structure may be inherent in the environment being modelled, or it may need to be created or emphasised by the VE developer. The combination of the fidelity and the enhancements of the VE can be used to give the user a good comprehension of the layout enabling them to orient and position themselves and ultimately to build a cognitive map of the VE. If the user is not present in the VE for long enough to warrant them building a cognitive map then arrows can be used to direct the user to their goal.

The author has always advocated the use of human viewpoints for VEs in which walking would be the normal, real world mode of navigation. A human viewpoint being one where the user moves around the VE as a walking person would, viewing the environment from about 1.6 metres above the ground. The collision boundaries of the objects in the VE should be modelled (within the limitations of the VR system) so that progress through the VE matches the user's expectations from the real world. Further to this, if a user is prevented from moving in the desired direction by a collision boundary, the object causing the collision should be easily visible so that the reason for the obstruction is clear. However, in the virtual factory users were given access to some viewpoints that were not subject to gravity or collisions (at VIRART we call this way of navigating 'ghost mode'). Unsurprisingly in retrospect, many users found these viewpoints difficult to use, especially considering that the physical input device used was the spaceball, which all naive users have found hard to get to grips with.

For each of these three case studies selecting the viewpoints has been done by pressing keys on the keyboard. In each case instructions were provided without which the users would not have been able to find the viewpoints except through trial and error. A recommendation that came out of this work is that where multiple viewpoints are

available, they should be accessible via buttons within the VE display, and that these buttons should provide some sort of cue as to the viewpoint they will take the user to.

In the author's experience from the case studies, context (rooms or surroundings) can aid navigation by giving people an understanding of the type of environment they are in, and a perception of the scale of objects and the distance between them. In this respect the context added to the VE is giving the VE environmental fidelity and some cues as to what activities are afforded by the VE. In the cases of the virtual ATM and the PC network card replacement VE, the environmental fidelity was substantially improved by the extensive use of textures. Beyond the limits of the modelled area of the VE the user is confronted with a blank and uninteresting space that none in the author's experience has felt the urge to explore. Most of the VEs developed by VIRART have been limited in their size to one or two rooms and so getting lost or not knowing where to find something has not been an issue. In the case of the virtual factory, which did have multiple floors and rooms, extensive (written) instructions provided the user with directions to the various activities provided. The instructions are therefore providing some of the enhancements required to compensate for the lack of fidelity of the physical interface. However the lack of an explicit structure to the factory (most rooms were of similar appearance, especially the colour) may have led to some of the navigation problems reported by the users. The PC network card replacement VE requires little navigation; the user is guided through what is required by on screen instructions and by the environmental fidelity of the cues provided by the objects in the VE.

3.6.2 Interaction

VEs have, of course, been typically defined in terms of having presence, autonomy and interactivity ((Zeltzer, 1992) and many time subsequently). Of these, and especially for industrial applications of desktop VR, interactivity is arguably the most critical attribute, as it clearly defines the difference between VR and other 3D modelling systems. Here the term interactivity is being used in a broad sense, to mean any action on the part of the user that results in a change in the VE. An obvious case is pressing a button to instigate a process, but interactivity can also include navigation, since as a user moves through a VE it updates and changes according to their movements and position. There are many computer applications that offer varying amounts of interactivity, but it is reasonable to suppose that VR technology can offer more representations of the types of interactivity

commonly encountered in real life. As with all facets of VE development, there will be limits on the amount and quality of interactivity available due to the technical limitations of any particular system and user interface. The VE developer has to work within these limitations and develop a VE that the expected user population will find valuable and usable.

All three of these case studies illustrated ways in which the design of points of interaction in a VE influences the usability of the VE as a whole. Of particular interest however is the Virtual ATM where, at the request of the client, little attempt was made to enhance the cues and feedback to aid the users' interactions. This went against the author's acquired instincts as a VE designer and prompted more thought about the whole issue of interactions in VEs.

From a VE developer's point of view it can be seen that it is important to think ahead when designing interaction into a VE. A decision has to be made, possibly on an object by object basis, whether to model the object 'realistically' including any real world design flaws, or whether to enhance the virtual object in some way to compensate for deficiencies in the VR - human interface. Ideally a user in a VE should generally feel that they are interacting directly with the virtual objects in the VE, whilst they are actually interacting with the devices that make up the VR system user interface (see section 4.2). Due to the limitations of current VR technology the user interface is likely to form a barrier between the user and the VE, whatever the nature of the sensors and effectors and whichever of the user's senses are engaged. One of the jobs of the VE developer is to make this barrier as transparent as possible within the constraints of the application.

When a user enters a VE it is normal that they perform some activities that affect the VE, otherwise they may as well be looking at a picture or watching a video. In order that they can successfully perform these activities the VE should offer some cues that tell the user what types of activity are afforded; in this sense VEs are no different to human-computer interfaces generally or to consumer products in terms of the concept of affordances (e.g. (Norman, 1988)). Cues could be in the form of navigational devices to aid the user's movement through the VE and to help them recognise their current location. Alternatively they could suggest to the user that a certain type of activity will afford a particular response. The user will benefit from knowing which objects afford interaction,

whether they are available for interaction at that moment, what method of interaction is appropriate, and what effect interaction with that object might have on the VE. Then having interacted with a chosen object, they will want to know whether that interaction was successful (see Tables 3.3 and 3.5).

	No interaction possible	Interaction not available at this time	Affords interaction
Little or no cue	user assumes correctly that the object is non-interactive	user may not attempt interaction, however if interaction is attempted, and with appropriate feedback, the user will realise the potential for interaction	user may not attempt interaction
Inappropriate cue	user may try in vain to interact with the object		unsuccessful interaction is likely
Appropriate Cue	user will correctly not try to interact with the object		successful interaction is likely

Table 3.3. Possible combinations of cue for, and potential for, interaction

	No interaction possible	Expected interaction not available at this time	Affords expected interaction
No immediate feedback	user may assume correctly that the object is non-interactive	user may not realise that interaction with this object can, but currently does not, have the desired effect	user assumes wrongly that the object is non-interactive
Inappropriate feedback	not applicable		user does not know what they have done
Appropriate Feedback	not applicable	user realises that this action can, but currently does not, have the desired effect	user knows what they have done

Table 3.4. Possible combinations of potential for interaction and resulting feedback

	No immediate feedback	Inappropriate feedback	Appropriate feedback
Little or no cue	user assumes that the object is non-interactive	user may not attempt interaction	user may not attempt interaction
Inappropriate cue	user may try to interact with object, but will not know whether that interaction was successful	unsuccessful interaction is likely	initially unsuccessful interaction is likely
Appropriate Cue		successful interaction may be aborted	successful interaction is likely

Table 3.5. Possible combinations of cue for, and feedback from, interaction

When the user attempts to interact with an object in a VE there are a number of possibilities for feedback, as shown in Tables 3.4 and 3.5. To enable a user to successfully interact with a VE, suitable feedback has to be present. Many of the potential problems outlined here can exist also when interacting with objects in the real world, but some may be exacerbated in a VE due to the limitations of the VR system hardware user interface.

During VE development, tables could be used to relate the interaction points in a VE with the cues, feedback and methods of interaction. For each interaction point within a VE a check could be made as to the cues and feedback available to the user in the various stages of interaction with an object. By looking at these factors in combination the developer can see which objects have adequate cues and feedback and which need to be enhanced in some way to enable successful user interaction. An example of a decision table for the 'virtual ATM' application is shown in Table 3.6.

The conceptualisations presented in tables 3.3 to 3.5 are designed to increase a VE developers understanding of the interaction requirements of a VE. During the VE development process they could be used as a reference tool. The decision table approach illustrated in table 3.6 could be less useful, as drawing one up during VE development could distract from the VE development process itself, and applying it retrospectively would also be time consuming and would only correct mistakes after they had been made. A better guidance tool could be one that leads the developer through the process step by step, prompting them to add extra cues and feedback where necessary.

Table 3.6. Decision table for interaction points - the case of a virtual ATM

Point of interaction	Cue to possible interaction	Method of interaction	Effect of interaction	Main feedback	Availability of interaction when object is visible	Feedback if interaction is currently unavailable
all room doors	implicit	collision	door opens	visual	always	n/a
Lock	hidden	mouse	console becomes unlocked	audible click	sometimes	none
Console	obscure	mouse	console opens if closed, closes if open	visual	sometimes	textual prompt
Mode panel	colour coded	mouse	slides out if in, in if out	visual	always	n/a
Mode switch	implicit	mouse	toggles display screen between normal and supervisor mode	visual	always	n/a
Outer door catch	colour coded	mouse	outer door opens	visual	sometimes	none
Outer door	implicit	mouse	outer door closes	visual	sometimes	textual prompt
Receipt printer release catch	colour coded	mouse	allows receipt printer module to be racked out	small visual	sometimes	none
Receipt printer handle	colour coded	mouse	racks out or replaces receipt printer module	visual	sometimes	textual prompt
Receipt paper feed switch	implicit	mouse	feeds paper through	visual	sometimes	none
Receipt paper spool spindle	colour coded	mouse	multiple effects depend on current state of system	visual	sometimes	textual prompt
Receipt paper	obscure	mouse	multiple effects depend on current state of system	visual	sometimes	textual prompt
safe door combination lock	implicit	mouse	unlocks safe door	none	sometimes	none
safe door handle	implicit	mouse	opens safe door	visual	sometimes	none
safe door	implicit	mouse	closes safe door	visual	sometimes	textual prompt
Currency cassette release catch	colour coded	mouse	allows currency cassette to be removed	none	sometimes	none
Currency cassette handle	colour coded	mouse	removes or replaces currency cassette	visual	sometimes	textual prompt
Currency cassette lid release catch	colour coded	mouse	opens or close currency cassette	visual	sometimes	textual prompt
Currency holder	colour coded	mouse	fills or empties currency cassette	visual	always	n/a

3.6.3 Applying Human Performance Theory

In the 'virtual ATM' case study, an environment that really exists was being modelled. In order to maintain an acceptable rendering speed it was necessary to carefully decide which objects to model from the many objects present in the room. It is when making these decisions that the world developer has to play God and decide what is good for the VE and what can be constructively left out. Bearing in mind the purpose for which the environment is being built it is possible to come up with a list of objects that are essential or very desirable to have in the model. It was decided that if it was possible to generically and comprehensively define and categorise the types of objects that might be needed in a VE, this would help VE designers to decide what objects to include. The categories defined by the author were; 'boundaries' (walls, ceiling, doors), 'landmarks' (doorways, pillars, prominent objects which give a VE an explicit structure, see section 3.6.1), 'obstacles' (any objects that impede navigation), 'devices' (tools, points of interaction etc) and 'features' that aid recognition (articles specific to that environment, which increase the VE's environmental fidelity, see section 3.6.1). It is possible for objects to belong to more than one of these categories. Objects not belonging to any of these categories can generally be safely omitted from a VE. Conversely, and ideally, objects belonging to one or more of these categories should only be omitted if they are either irrelevant or detrimental to the purpose of the VE, although this can be difficult to determine. What actually happens is that sometimes relevant, non-detrimental objects are also omitted due to a shortage of development time or because they are just too difficult to model. It was at this point that it was decided that some understanding of human performance theory could help to decide which of the categorised objects should be included in the VE.

The goal of the VE world developer is to create effective, safe, usable virtual environments that enable client organisations and their users to achieve their goals in a better, more motivational and cost-effective fashion than by using alternative technologies or by using the equivalent real situation. As suggested in section 2.7.3, producing VEs that do this should ideally be based upon an understanding of how people behave and carry out tasks within VEs; developers need general guidance on human performance within VEs and specific guidance on enhancing VE usability. Clarity about

what determines usability of VEs will be enhanced with some better understanding of participant performance.

Storyboarding should outline the VE specification and how it should be designed; however, some identification and categorisation of participants' requirements will be needed. This can be achieved by task and user analyses and virtual task analyses. A more detailed examination should be made of how participants will respond to different elements of the VE, utilise all its functionality, and be able to comprehend and use all the interface elements to meet the application goals. As part of this, developers need to know how to encourage participants to explore the VE and enable them to understand which elements can be interacted with, identify how this interaction might be achieved, and minimise dysfunctional participant behaviour and serious errors. In other words an improved understanding of human behaviour and performance within a VE is needed, but such understanding is in its infancy.

There are a number of candidate theories and approaches that might help. For instance, behaviour of groups of workers in complex technical and organisational environments can be interpreted within a framework of distributed cognition (e.g. (Hutchins, 1995)). In such a view, the handling of information and decision making which characterises work in transport control, for instance, is distributed in time and space across teams of people and various computer and telecommunication systems. An ethnographic or similar approach may then be taken to study and interpret work behaviour and skills. This may be a framework within which to study performance in networked VEs or collaborative VEs (CVEs).

Another possibility is to build understanding of participation in VEs, and thus guidance for their design, around the notion of situated action (Suchman 1987). In this view, work cannot be understood merely in terms of individual skills and institutional norms, but is the product of continually changing people, technology, information and space. In this view, people do not approach tasks with a clear set of plans in mind, but build and modify their plans in the course of situated action.

A third possibility for understanding performance in VEs is to utilise the notion of situation awareness; people perceive cues from their environment and use them to make

sense of the current state of the world and to project this understanding into the future (Endsley, 1995). Various measurement methods are available to assess situation awareness that might have value in understanding and measuring performance in VEs. Also, there is a strong connection between the role played by situation awareness and the participant's mental model. Guidance on design usability then, would concentrate upon those VE elements expected to help build a strong and "correct" mental model for the participant or to match their existing ones, and to contribute to high levels of situation awareness.

Finally, help might be found in the large variety of models and frameworks used to understand, predict and reduce human error, generally in safety critical systems. A particular possibility here is Rasmusson's (1986) well-known SRK model of skill-, rule- and knowledge-based behaviour. If the types of tasks in the VE can be defined as to whether they are skill, rule or knowledge-based it might be possible to distinguish between the types of design guidance offered for each. More usefully, characterisation of real-world tasks as skill, rule or knowledge-based, and particular variants of these, could allow association of different ways to deliver their representations in the VE. The SRK model is often used in conjunction with Reason's (1990) typology of human error – slips, mistakes and violations – and the underlying psychological error mechanisms involved. Again, adaptation of these to the VE domain could allow a better understanding of the types of behaviour exhibited by VE participants and, subsequently, lead to production of useful development guidance.

Skill-based behaviour is shown in tasks where a signal brings recognition of a required response, such as adjusting a control to keep a display at a certain level. The need here may be for the virtual display to provide signals that are quickly and clearly recognisable and for the response actions to be natural and easily associated with the signals. In other cases the design guidance may make clear that some sensori-motor skilled activities will have to be represented in the VE by a metaphor, for instance pointing at, and selecting, a part of an object rather than "picking it up" with haptic feedback. It should be possible both to model real world rule-based behaviour in a VE, and also to teach rules in a VE which can then be applied in the real world. For example, if a user is in a VE and is instructed to open a door, they will already be in possession of several "rules" which will help them in the execution of this task. In this case the rules might be; "If the object is an

upright rectangular shape in a wall, then it can be opened to pass from one space to another", "If the door has a small shape about halfway down to one side, interact there to open the door" and "If the door opens, walk through". Knowledge based behaviour is used in problem solving situations. A mental model of the situation is formed, and this model will be used as a basis for a plan of actions. However, if a user is presented with misleading or inadequate information, then they may be unable to form a plan, form an incorrect plan, or may form several plans and be unable to determine which of these will be successful. Advice on design and content of the VE can underpin the user developing a correct mental model, and therefore being more likely to adopt a successful strategy in the VE (or in the real world once they have been trained in a VE). This can be aided by making sure that the options available to the user are restricted to those that are relevant to them building knowledge appropriate to the goals of the application. This availability of options will in turn be determined by what objects are modelled in the VE. It is therefore in this area of knowledge based behaviour that this theory can offer most help to the process of object number reduction; the VE developer should prioritise the modelling of objects that support and add meaning to the users' activities in the VE. To this end, the current list of objects necessary in a VE (boundaries, obstacles, landmarks, devices and features) can have four more types of objects added to it. (1) Cues – any objects that indicate affordances, e.g. an on/off switch. (2) Scale informants – objects that have a (approximate) set size regardless of their context (e.g. a chair), and that can therefore be used (subconsciously) by a viewer to gauge the scale of neighbouring objects. Not all scale informants need to be modelled, in many situations one or two give sufficient information. (3) Pathways – objects providing a continuous visual reference, showing the route from one place to another, e.g. a road or a handrail. (4) Aids – anything put into the VE to provide information not immediately implicit from the geometry, e.g. a list of instructions, map, signpost etc.

3.7 Conclusions

Most VE development, and especially the development in the three case studies documented in this chapter, is about maximising the functionality, effectiveness, usability and likeability of the VE whilst minimising the cost, most of the cost being VE development time. During these three case studies, and building on the experiences of the first five case studies, techniques have been developed to allow developers to build VEs with these attributes in a more efficient way. Most of these techniques fall into three main categories; navigation, interaction and object number reduction. However, all the techniques impact on each other in various ways, and so an overall structure is needed within which these techniques can be placed, so that they can be applied in a more systematic way.

Chapter 4 A More Comprehensive Structure for the Development of VEs

4.1 Introduction

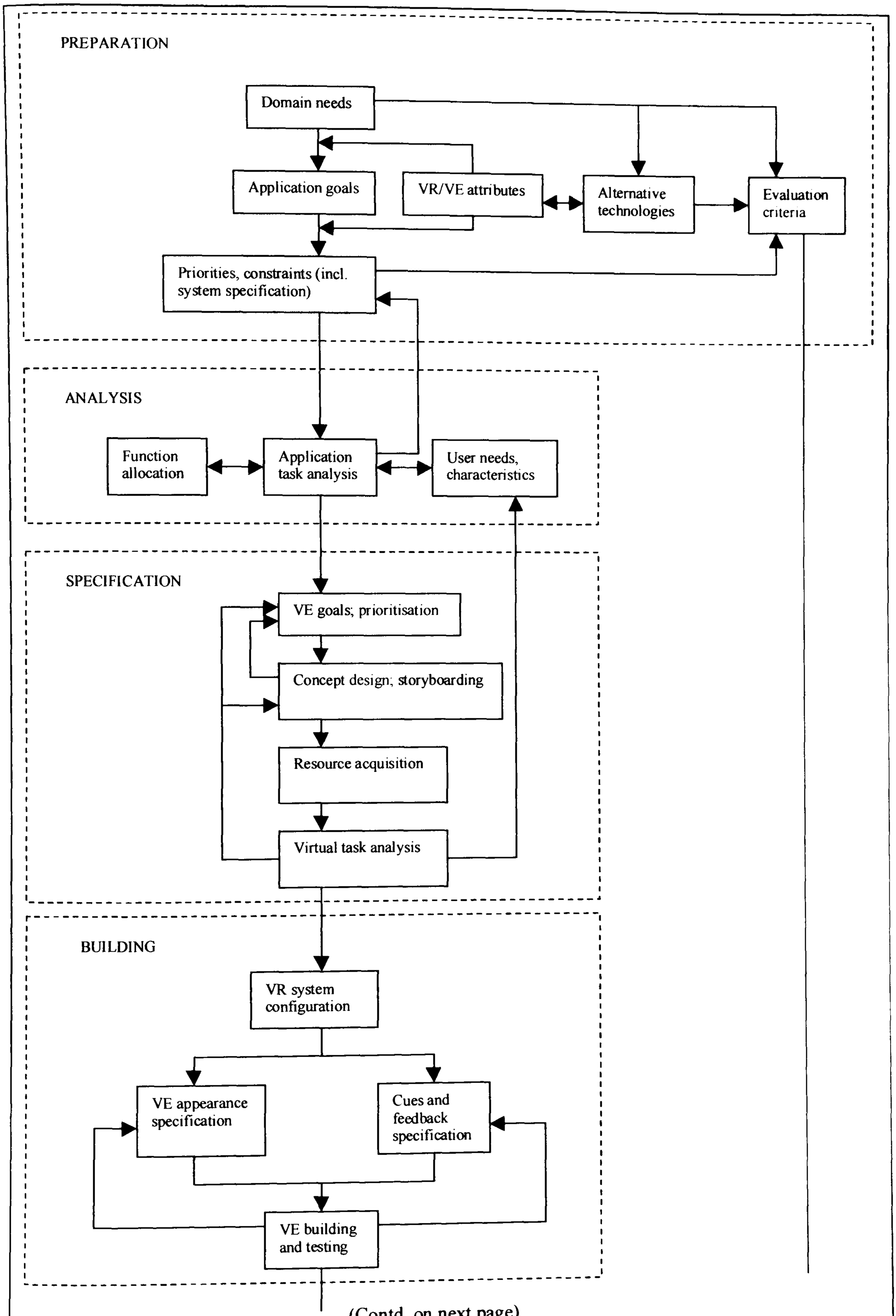
To some extent, VEs could be developed following design process guidance for any human-machine system or human-computer interaction, of which there are vast numbers (e.g. Preece, 1994, Newman and Lamming, 1995; Sanders and McCormick. 1997; Sutcliffe, 1995; Wickens et al, 1998). However, the effort required to model the objects and actions of the real world and the nature of the user cognitive interface (the participant being 'within' the database) mean that VE development also requires guidance specific to its own special characteristics and needs. In any case, guidelines and models within the human factors and "conventional" HCI communities are notorious for being not always useful or usable by designers, engineers or even ergonomists; acceptable and useful guidelines to design for usability are not easy to produce in ergonomics generally. We might, of course, draw upon any similar development guidance in simulation design, but VEs have sufficiently different attributes and purpose for its own structured development framework and associated guidelines to be needed. Finally, whilst the computer games sector has developed numerous libraries of routines for the programming of features commonly required by games developers, there have been no attempts to provide an overall structure for games development from concept through to finalising the product. It was therefore decided that the Virtual Environment Development Structure (VEDS) should be adopted with a view to refining it in the light of experiences gained during the subsequent case studies.

4.2 Developing VEDS

The first version of the structured framework for building and testing VEs was drawn up in 1994 from the experiences of the author and colleagues in the development and evaluation of VEs, especially the virtual factory (see section 3.3) (Wilson et al., 1996). As each subsequent VE developed was evaluated, and the development process was reviewed, this framework was continually refined (D'Cruz, 1999; Wilson, 1997) and the version shown in figure 4.1 was the one in use in 1998. It was developed from the work documented in this thesis and other work outside the author's sphere of influence, and from the contributions of a number of researchers from different backgrounds, none of

whom individually had 'editorial control' over the final form of the structure. A full description of its development was published by Wilson et al (2001).

This version of the structure divides the development of VEs into 6 sections, preparation, analysis, specification, building, implementation and evaluation. Whilst all the sections have influence on this thesis; specification, building and implementation are the sections most likely to be influenced by this doctoral research, and at this stage only an outline structure has been provided for the design and building of VEs. By applying this structure to the development of some case study VEs it is hoped that, as well as aiding the development of the VEs, it will be possible to evaluate, validate and refine the structure itself. In particular it is hoped that more detail can be added to the specification, building and implementation sections.


PREPARATION
Domain needs
Application goals
VR/VE attributes
Alternative technologies
Evaluation criteria
Priorities, constraints (incl. system specification)
ANALYSIS
Function allocation
Application task analysis
User needs, characteristics
SPECIFICATION
VE goals; prioritisation
Concept design; storyboarding
Resource acquisition
Virtual task analysis
BUILDING
VR system configuration
VE appearance specification
Cues and feedback specification
VE building and testing
(Contd. on next page)

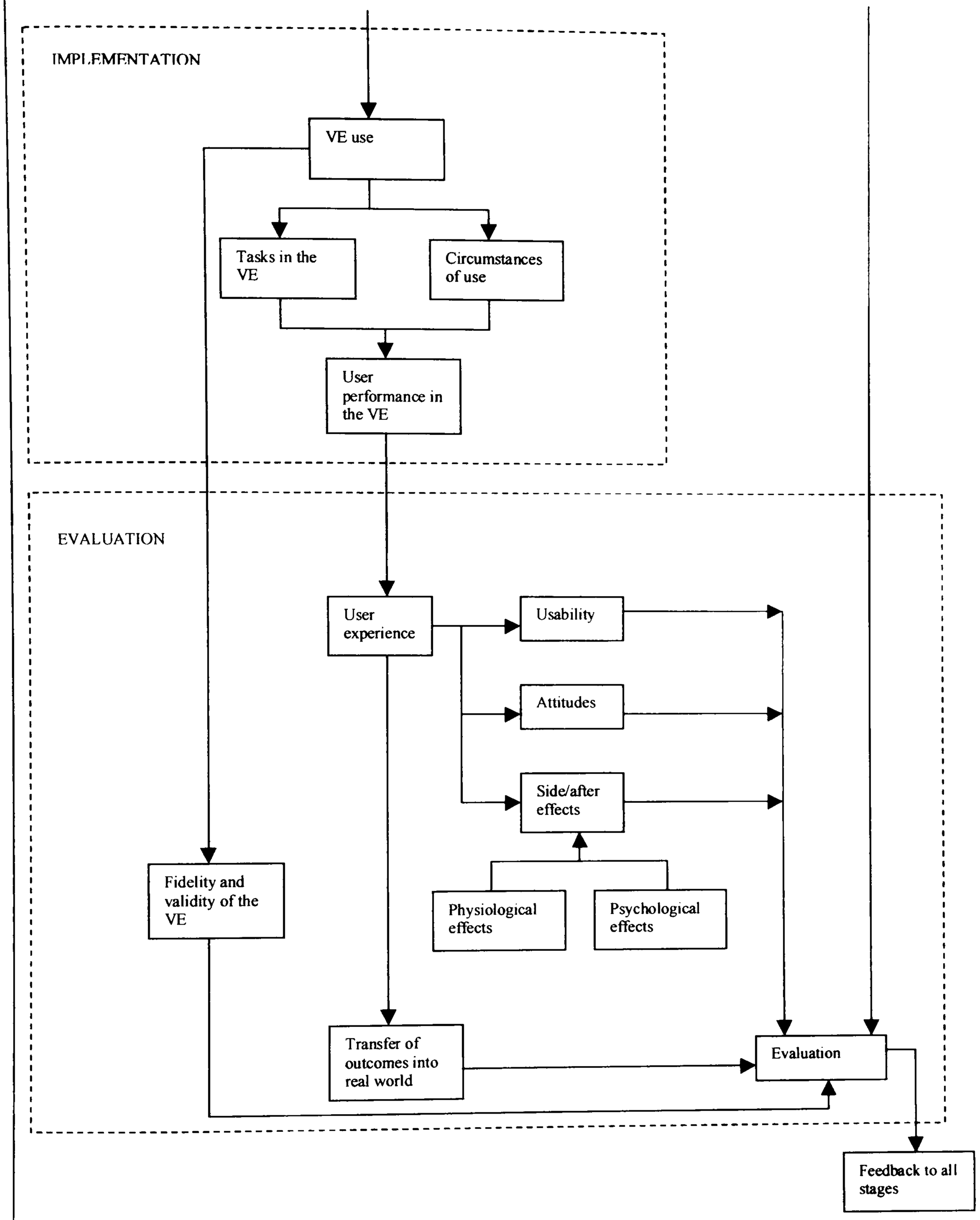


Figure 4.1. VEDS: Virtual Environments Development Structure [Wilson, 2001 #18]. Note: only major feedback loops are shown for clarity; in fact the whole process is an iterative one.

Chapter 5 Implementing VEDS

5.1 Introduction

Chapter four (section 4.2) describes VEDS and explains how specification, building, and implementation are the areas most relevant to this doctoral research. In this chapter these sections of VEDS are applied to the development of two VEs, one for research into the use of VEs for training, the other for research into the use of VEs in mainstream education. A summary of the case studies can be found in table 5.1.

	Project name	**Collaborating bodies**	**Description of application**	**Main design issues**
Case study 5	Building with virtual Lego	EPSRC	Research into the use of VR for training using building in Lego as the task	Usability, transfer of training to the real world
Case study 6	RadLab	EPSRC PPARC	Teaching radioactivity at key stage 4 of the UK national curriculum	User centred design, consistency of metaphors

Table 5.1. A summary of the case studies examined during the exploration of development issues

In each of these case studies the description of the development of the VEs will follow the sequence of events as laid out in VEDS. Where the actual development process followed during a case study varies from VEDS this will be indicated in the text.

5.2 Case study five: Building with virtual Lego

5.2.1 Introduction

This VE was developed as a follow up to the PC network card replacement VE reported on in case study 4 (section 3.5), to further investigate the use of VR in training. The project was the continuation of the same PhD research of a colleague into the evaluation of training in VEs (D'Cruz, 1999) and the VE design and development was therefore undertaken collaboratively. Whilst the results from the earlier case study highlighted some important issues relating to the use of VET, some aspects of the task trained (e.g. the use of a screwdriver) obscured significant differences in the effectiveness of the training. This was further compromised by the level of skill required to navigate using the main input device (the spacemouse). Therefore this new study was devised to

examine the use of VR to train a different task requiring less specialist skills, and using the standard mouse as the main navigation device.

Preparation and Analysis

The preparation and analysis stages of this case study are more fully documented by D'Cruz (1999). Going through the preparation and analysis stages of the framework resulted in the proposal of a VE to train a user to build a Lego model of a motor vehicle (this task was chosen as it would not require any specialist skills). Three groups were to be trained to perform the task; one using an interactive VR training package, the second, a passive video training package, and the third group were to be given no training. The performance of the trainees from each group could then be assessed when building the Lego model in the real world. To further reduce the variables in the experimental method the 'video' training package was in fact to be a passive (animated) version of the same interactive VR training package. Effectively, it would be a 'video' delivered by a VR system. This would ensure that there would be no differences in the way the packages were viewed (such as screen clarity) to affect the results. A major limiting factor on the development of this VE was that the design, building and testing of the VE had to completed in two months in order to fit in with the research schedule.

5.2.2 Specification

VE goals; prioritisation

The overall goals of the VE application were defined as;

1) To train subjects, with no expertise in VR, to follow a procedure to assemble a toy car.
2) To highlight specific features of desktop VR/VEs that may be beneficial to training applications.
3) This to be done with a view to comparing VET with other methods of training.

From the VE developer's viewpoint the priority was to design a package that could be used effectively with minimal training, by naive users. Responsibility for the inclusion of features to fulfil the other goals rested largely with the collaborator in the project.

Concept design; storyboarding

One of the project's aims was to highlight specific features of desktop VR/VEs that may be beneficial to training applications. It was therefore a prerequisite that we use a desktop

VR system. The system used was a Pentium 133 PC, a 15-inch colour monitor, sound card, speakers, keyboard and mouse, running Superscape VRT 4.00 software. Whilst this configuration was largely determined by what was available at the time, it can also be argued that this was a good choice of system for the reasons of; ease of programming (using local expertise), low cost and availability of platform, and expected utility of the resulting application.

For the integrity of the research, this evaluation of VET had three scenarios that needed to be kept consistent with each other, these being; the interactive training, the passive training and the real world application of the skills acquired. To achieve this it had to be ensured that whatever was built in the VE could be replicated in the real world, and conversely, that everything that was put into the real world scenario could be built satisfactorily in the VE. This was achieved by setting up the real world scenario first, and including a minimal number of items, i.e. a table with the Lego components laid out in a parts grid (see figure 5.1). To keep the user interface simple and remove the need for the user to learn how to use the VR system, navigation would be reduced to moving between three fixed viewpoints. The storyboarding for this project was textual, a summary of it follows here along with some of the reasoning behind the decisions made.

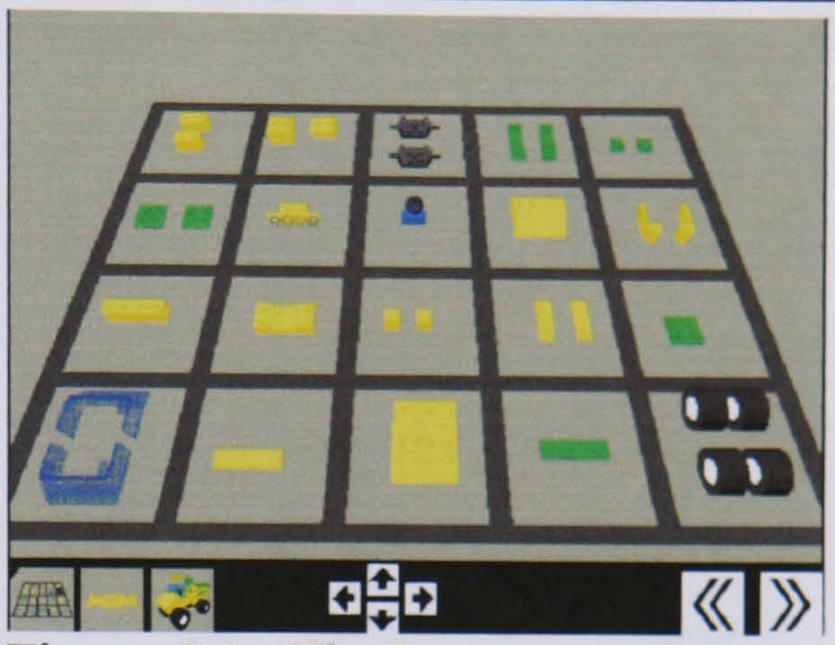


Figure 5.1. The Lego components laid out on the parts grid

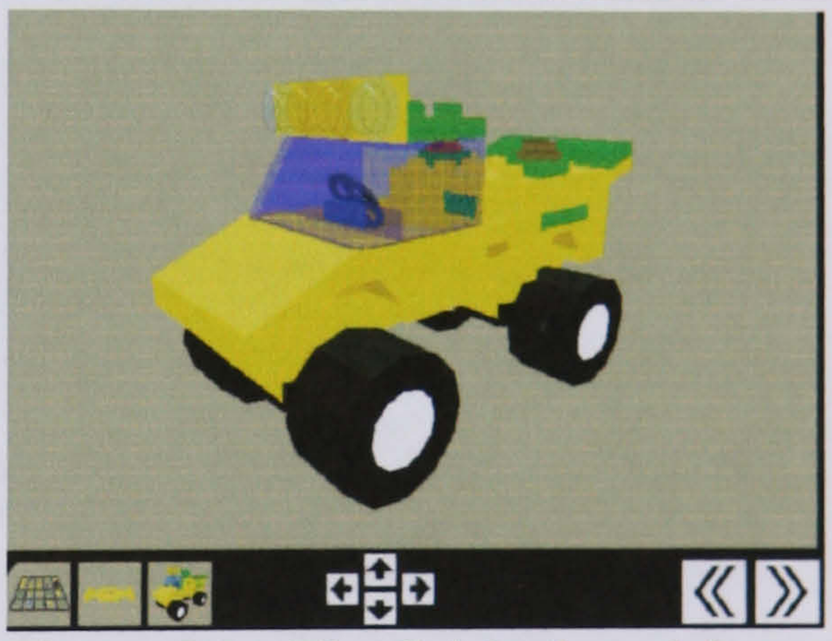


Figure 5.2. The finished model

In the real world the user could, by moving their head, choose to look at the parts grid, the picture of the finished model, or the partially completed model that they were working on. This facility was to be made available to users in the interactive training VE by clicking on the buttons at the bottom left of the display (see figures 5.1 and 5.2), with the added bonus, that the picture of the finished model was to be a 3D representation of the finished model. In the real world the user could pick a component from the grid, add it correctly or incorrectly to the model, remove it and place it again, and then select the next piece.

To simplify the development process and allow VE building to fit into the tight development schedule, it was decided that the user should be restricted to picking the components up in pre-defined sequence, and that a component would only attach to the model if the user clicked on the correct 'stud' to attach it to. Clicking on an incorrect 'stud' elicited no response. This meant that if the user attempted to place a component in the wrong place they would get no feedback. Whilst this might initially cause the user some confusion it was felt that this would soon be dispelled by the repetitive nature of the picking and placing task, and that an error message (visual, audio, or both) would quickly become annoying as, by the nature of the task, mistakes could be made frequently. In the real world the partially assembled model can be rotated and viewed from any angle. To facilitate this in the VE, four buttons with arrows on would be arranged at the bottom of the display (see figure 5.1). Clicking on any of these buttons would rotate the model in the direction indicated. If the viewpoint onto the finished model was selected, then this finished model would be rotated instead. In the real world the user can, at any time, decide to remove components from the partially completed model, and then rebuild it. This would be made possible in the VE by providing a 'go back' button, each click on which would remove one component (most recently added, first removed). To aid comprehension, this button would have the same symbol as the rewind button on a VCR or cassette player (see figure 5.1). A feature to be made available in the VE, but not available in the real world was the button that automatically picked and placed the next component in the sequence. In keeping with the 'go back' button this would bear the symbol for fast forward (see figure 5.1). Thus the user in the VET could at any time automatically step forwards or backwards through the construction of the Lego model, then at any point start assembling it 'manually' again.

This storyboard for the interactive VE was then adapted to become the storyboard for the passive 'video' training package by removing all the facilities for user interaction and replacing them with a timed automatic application of the 'fast forward' button. This was to be combined with an automatic rotation of the partially completed model to allow best viewing of the assembly procedure. Thus the user would get a good but pre-programmed view of the automated assembly of the model.

Resource acquisition

The resources required for developing this VE (not including computer equipment) can be split into two categories. The first category includes the items required to make up the training scenario, i.e. the Lego components and the table. These items needed to be acquired at this stage in order to finalise the storyboard and so that the subsequent task analysis could be carried out. The second category of resources includes all the dimensions of the components of the scenario (in this case the table and Lego), recording their appearance (shape and colour) and where appropriate, taking digital photos of the components for use as textures within the VE. Whilst the first category of resources was important for this project, it would not normally be required during the development of a VE. This project was an exception, as it required a real life scenario to be assembled, on which the VE was then modelled. The second category of resources will be common to most VEs. However, in this case study this resource acquisition did not take place at this stage. Instead it happened as part of a more detailed VE design process later in the VE development.

Virtual task analysis

A task analysis of the real world process of building a Lego model had taken place during the 'analysis' phase of the VE development process for this VE (figure 4.1) and is documented by D'Cruz(1999). In order to ensure that the VE being designed will be usable by the target population it is necessary to carry out a virtual task analysis. This will be largely based on the 'real' task analysis, however all the actions will now be carried out via the hardware and software user interface of the VR/VE system. Some actions that were relatively simple in the real world may become very complex in the VE with the available modes of interaction. In this case study one of our primary aims had been to develop a user interface that required very little training and skill to use, but which adequately modelled the (cognitive) process of building a Lego model. The repetitive nature of the actions required allowed us to simplify the virtual task analysis to a list of the generic actions, as follows;

1	Look at finished model	Click on finished model icon
2	Look at partially completed model (at this stage just a bare chassis)	Click on partially completed model icon
3	Look at component grid	Click on component grid icon
4	Pick up first component	Click on first component
5	Look at finished model	Click on finished model icon
6	Rotate finished model to get best view	Click on arrows at foot of screen
7	Look at partially completed model	Click on partially completed model icon
8	Rotate partially completed model to get best view	Click on arrows at foot of screen
9	Attempt to fit first component	Click on stud(s) on partially completed model until part clicks into place
		Repeat from the beginning

Table 5.2. The virtual task listing, showing a list of the generic actions.

As can be seen from the table (5.2), all of the interactions are implemented by the user clicking the mouse on the required icon or virtual object, making this the only skill required when using the VE.

5.2.3 Building

VR system configuration

The VR system configuration for this project was first considered during the preparation phase of VEDS. This process was finalised during the specification stage as part of the concept design.

VE appearance specification

In this case study the VE appearance had been specified during the concept design and storyboarding phase of the VE development. There was however, some work to be done deciding on the level of detail to be applied to the Lego components. This in turn brought about resource requirements, such as texture maps of the underside of Lego bricks that were deemed too complex to model realistically using geometry.

Cues and feedback specification

In this case study, as the cues and feedback were fundamental to the objectives of the VE, they were fully specified at the storyboarding stage. This would not always be the case however, with many VEs having a much less detailed earlier specification.

VE building and testing

Before the actual building of the VE could start, an extensive period of detail design needed to be done to decide how the object behaviours and functions would be made to work correctly. When a Lego model is assembled, components added to the model fit over and around the 'studs' of the components already assembled. Implementing this in Superscape would cause overlap problems as the 'studs' of one object would be inside the cuboid shape of another object. Further to this problem, whereas before assembly the components move separately, the now assembled components have to move together as if they are a single object. In Superscape this requires the added components to become part of the group that includes the assembled components. Whilst it would have been possible to write algorithms to overcome these problems and at the same time maintain a 'literal' model of the assembly process, it was decided that there was a simpler and

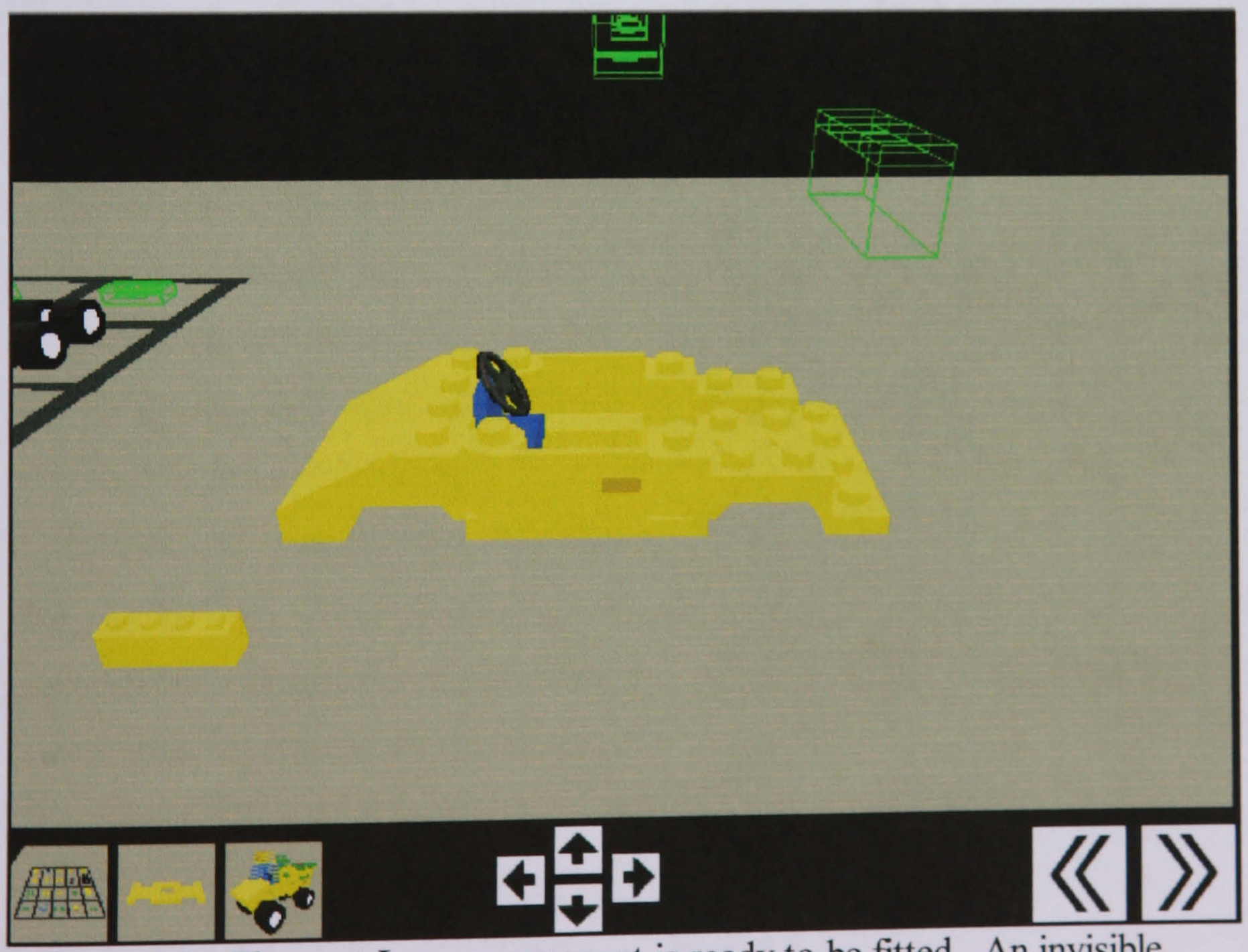


Figure 5.3. The next Lego component is ready to be fitted. An invisible copy of the component is positioned over its final position.

quicker method that would result in a more robust model. This method involved building a separate virtual Lego model for each stage of the assembly process. Each of these models would show an animation of the components for that stage moving towards and fitting onto the model. This animation would start with the new component(s) on the table near the partially assembled model. At the same time, duplicates of those component(s) would be positioned near their final position on the model. Both the component(s) on the table, and the ones near the model would initially be invisible. Once it/they were selected from the grid, it/they would become invisible on the grid and visible on the table near the model (figures 5.3 and 5.4, outlines of invisible objects shown in bright green for this explanation only). When the user clicked on the correct stud for a component to fit to, that component would become invisible on the table and visible near to its final position on the model (figure 5.5). It would then move toward that stud and click into place (figure 5.6). As soon as the component(s) for that stage was/were fitted, the model would be replaced by the model for the next stage. This replacement was achieved by making all the models invisible except for the one required for the current stage of assembly (to keep them 'physically' separate the invisible models are positioned behind the table and can be seen in the background in figures 5.3, 5, and 6). The grid has 20 (groups of) components in its 20 spaces. This equates to 20 construction stages, plus

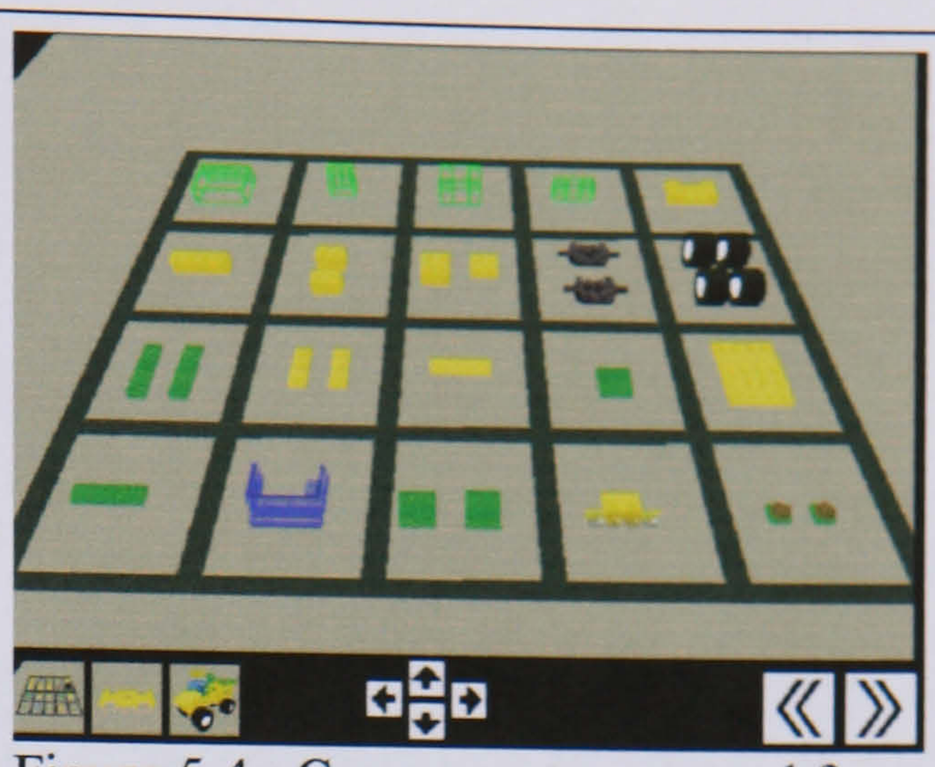


Figure 5.4. Components removed from the grid become invisible (shown here in bright green).

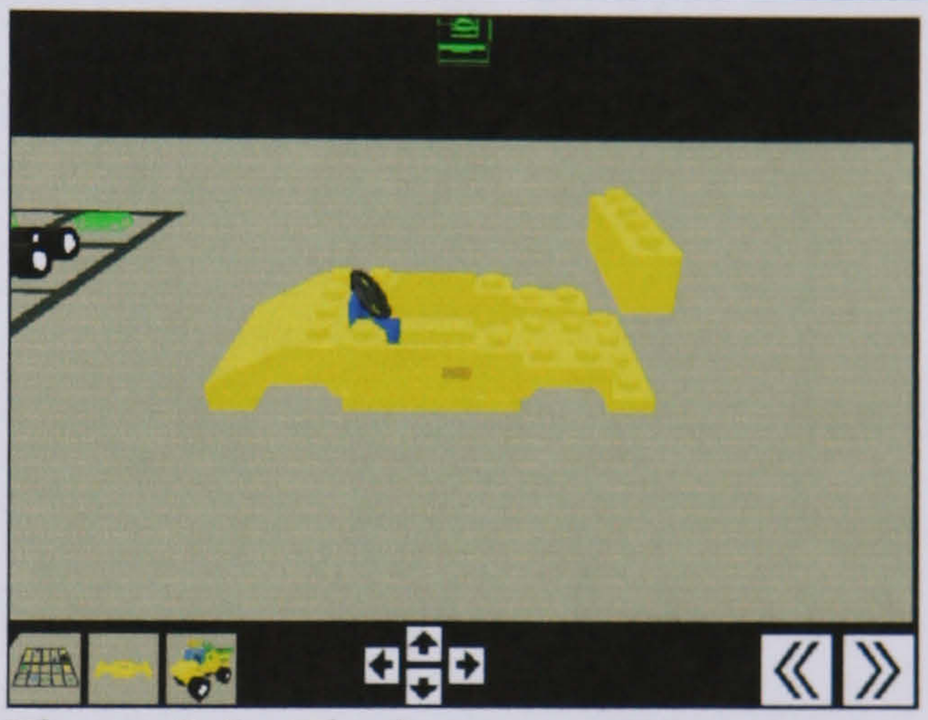


Figure 5.5. The Lego component moves towards its final position.

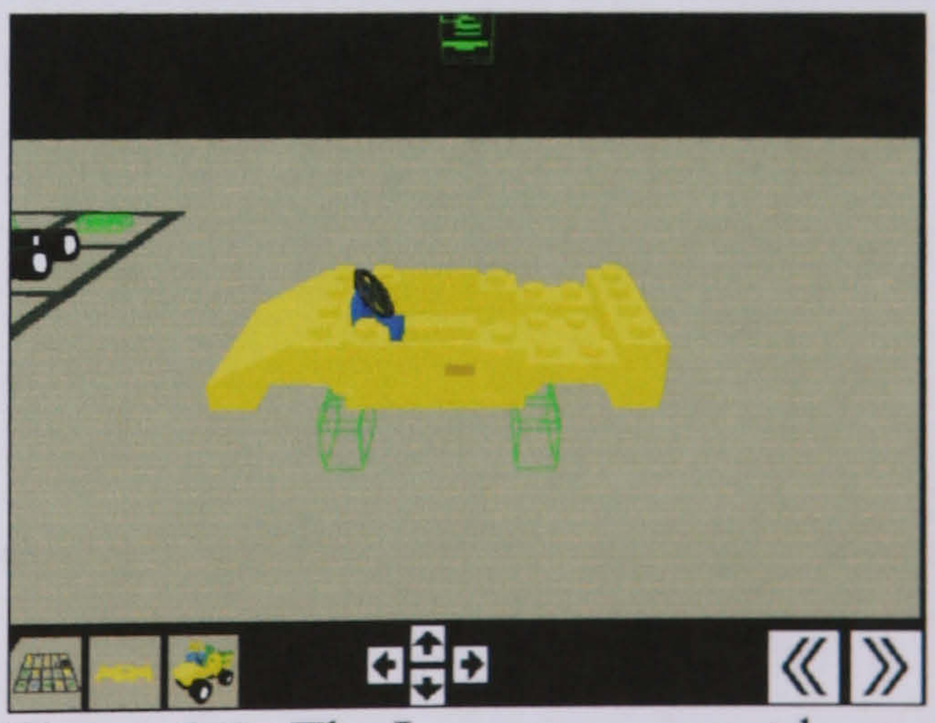


Figure 5.6. The Lego component has fitted into place, and the model has been replaced by one for the next stage of the assembly procedure.

the starting point of an empty chassis, which makes 21 models in total. The problem of overlapping objects described earlier was overcome in these models by not including the hidden studs (that would cause the overlaps). Also assembled components would move together as a single object because they would all be members of the same group throughout the entire process. No changes in group membership would be necessary.

The 3D modelling process for this VE was inherently straightforward due to the blocky nature of the Lego bricks themselves. The curved surfaces, such as the Lego 'studs' and the wheels were simplified to 12-sided cylinders. This kept the facet count down such that no further level of detail control was necessary. As the 'studs' were to be removed in the versions of the model where a brick was placed over them, the bricks were actually made separate to the 'studs' that were added and removed as necessary during the VE assembly process.

The assembly of the 3D shapes to make the VE was mainly taken up with the creation of the models of each of the 21 stages of the Lego assembly. The animations of the relevant Lego component moving into place were also added to each of the models at this stage. On each model the 'studs' to which the next brick should be attached needed to be programmed to activate the animation when clicked on by the user. Realism was enhanced by texturing the bottom of the Lego bricks where they would be visible, and by adding a clicking sound as each component snapped into place. The screen layout was set up with the seven buttons specified in the storyboard, three of the buttons selecting the different viewpoints, two buttons rotating the model, and 2 buttons programmed to step forwards and backwards through the assembly process.

There were three requirements on this VE from the specification; these were navigation, interaction and fidelity/validity. Firstly, when using the VE it had to be possible for the user to navigate freely to the parts of the VE necessary to complete the tasks. Secondly, it had to be easy for the user to understand what objects afforded interaction, how those objects could be interacted with, and what the results of that interaction were. Finally, the VE had to adequately represent the scenarios as per the VE specification, i.e. having visual and functional fidelity/validity.

There were two levels of testing for this VE. The first was the testing done by the developer on each new feature as it was created in the VE. This testing is against the criteria set out in the specification. The next level of testing was the pilot study stage where naive users were observed going through the VE carrying out the tasks, so that any observed or reported problems could be addressed.

5.2.4 Implementation

The implementation of this VE took the form of a series of experiments, the aim of which was to evaluate the use of desktop VEs for training. These experiments are only summarised here as they are fully documented elsewhere (D'Cruz, 1999). To reduce the external differences between the users' experience of the VE training, the 'video' training and performing the tasks in the real world, all the activities took place in the same laboratory. There were 36 subjects, 18 male and 18 female, aged between 22 and 44 years. Each subject took part in two sessions 5 days apart. Each session lasted one hour and consisted of a training component (this was omitted for the control group who had no training) either using VR or 'video', and the real world Lego assembly task. The subjects were videoed, and filled in questionnaires to get their opinions of the comparative merits of the various training methods.

5.2.5 Evaluation

In terms of performance in the Lego assembly task, both the users trained by 'video' and the users trained by the VE performed significantly better than the group with no training. There were no significant differences in the performance of the 'video' and VE trained groups. However, the VE trained group had a more positive response to their training medium and felt more motivated by it. From the questionnaires they *agreed* that the VE was, relevant, helpful, effective and useful. At the same time they *disagreed* that the VE was inappropriate, too slow, unhelpful and confusing. Features of the VE that the users particularly liked were using the mouse to rotate the model and step forward and backwards through the process, the general screen layout and the position and appearance of the buttons, the appearance of the virtual objects and the use of different viewpoints. Some users thought there should have been more interactions and that they could have been more realistic, although they did not think this detracted from their training. They did not have any problems interacting with the virtual objects and felt that this helped them perform the real world task.

The original aims of this research project were;

1) To train subjects, with no expertise in VR (or Lego), to follow a procedure to assemble a toy car.
2) To highlight specific features of desktop VR/VEs that may be beneficial to training applications.
3) This to be done with a view to comparing VET with other methods of training.

This VE was successfully used to train subjects to assemble a Lego car. The results also showed subjectively which features of VEs the users felt contributed towards them learning how to do the task. There was however, no objective comparison of the relative merits of different features of VE training applications. This VE was successfully used to compare VET with other methods of training.

5.3 Case study six: Teaching Radioactivity

5.3.1 Introduction

Preparation and Analysis

The preparation and analysis stages of this case study are more fully documented by Crosier (2000). The design and evaluation of this VE formed the major part of an EPSRC funded PhD research project (Crosier, 2000) looking into the use of VR in science education. The hypothesis was that VR could be useful in science education as it affords the presentation of complex concepts in 3D and from different perspectives. Radioactivity was seen as an appropriate area to pursue for a number of reasons. In the past radioactivity was taught in British schools by carrying out an experiment using radioactive sources and shielding materials. Health and Safety legislation has meant that the subject can no longer be taught in this way, so now teachers have to use other methods such as "chalk and talk" or videos. It was thought that by creating a VE to simulate this experiment, some of the "hands on" learning could be put back into the topic. In addition, the VE would provide extra capabilities such as allowing the students to zoom into the radioactive materials and allow them to view the atomic structure and the radioactive particles in three dimensions. As well as assessing the utility of VR for this type of teaching, a further aim of this research was the comparison of directed and non-directed study within a VE.

5.3.2 Specification

VE goals; prioritisation

The overall objective of the VE project was defined as; to specify and iteratively develop a VE for teaching the specified area of study, within a user centred design process. As with the 'Building with virtual Lego' case study, from the VE developer's viewpoint the priority was to design a package that could be used effectively with minimal training, by naive users, in this case 14 and 15 year old school students. This would require a highly interactive VE that, at the same time had a very simple user interface. The VE would also be required to incorporate different ways of viewing the same concept (naturalistic and schematic). Responsibility for the achievement of the other research goals rested largely with the author's collaborator in the project.

A user centred design approach was used, with teachers involved throughout the process as design partners to develop a prototype VE until it was suitable for first-stage school use. Following this, an iterative design-build-evaluate-redesign process was employed with school students and teachers involved in testing the VE and evaluating it, with the feedback being assessed at each stage and incorporated into the redesign of the VE.

Concept design; storyboarding

After initial contact with schools, and a teacher familiarisation process of VR/VE demonstrations, teachers were able to suggest a large number of areas for potential VE applications. The area of a virtual radioactivity experiment was selected as an area in which it was particularly appropriate to develop a VE.

The process of deciding on the target VR system configuration for this project consisted of establishing what computer systems were commonly available in the schools where this VE package was to be tested, and then deciding whether the specification of these computers would be adequate for our purposes. The teachers had requested an application that could be implemented on existing equipment installed in their school's computer laboratories. At the time the best of this equipment ranged from 100Mhz-200MHz Pentium PCs, although plenty of older PCs were available. We decided that it would be possible to build a VE that would run satisfactorily on a Pentium 100 with 16MB of RAM and an SVGA colour monitor with 800x600 screen resolution, running Microsoft Windows 95, so this was established as our minimum target specification.

For the purpose of drawing up a storyboard, a user group was formed comprising teachers, programmers and a researcher. Firstly, based on the teachers' comments that had requested minimal need for movement within the VE, it was decided that it should consist of a single room. The room should be made to look like a classroom or laboratory, as the teachers had liked this idea and also this put the learning into the appropriate context. The Geiger counter should be modelled and should be visible in the VE so that the students would learn about the appearance of the apparatus as well as its operation. The equipment should include a meter with a dial from which the radioactivity could be measured and sound should be added to give additional feedback and make the Geiger counter seem more realistic. The experiment should be modelled realistically with stands

for the equipment, so that the students could acquire the experimental skills associated with using apparatus and materials. Also, the radioactive materials should be made to

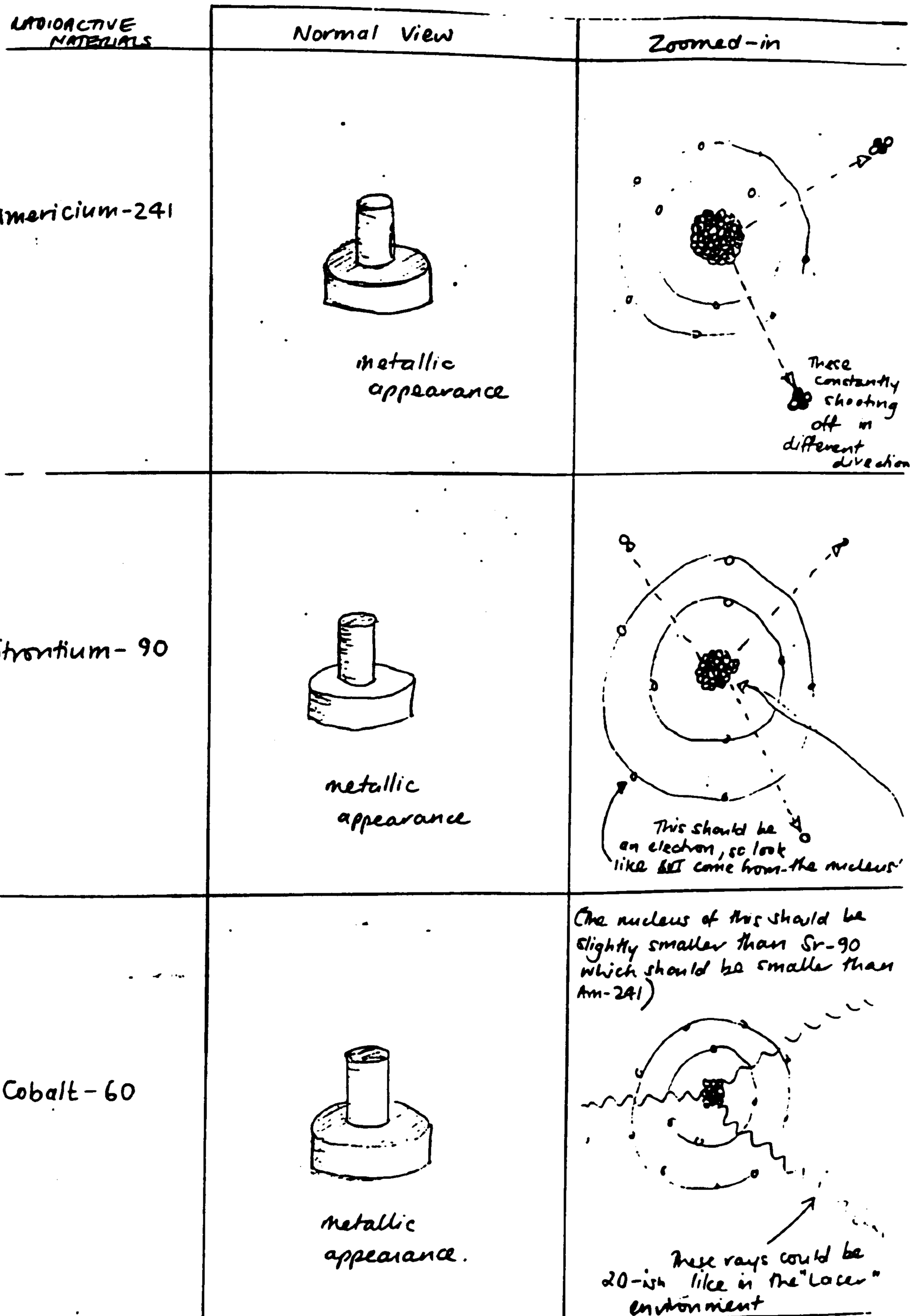


Figure 5.7. An excerpt from the storyboard for the 'teaching radioactivity' VE.

look like they would in a real experiment. A zoom facility that the teachers had wanted was incorporated into the design. An excerpt from the storyboard is shown in figure 5.7.

Resource acquisition

It was pointed out in the 'Building with virtual Lego' case study that some of the resource acquisition did not take place at this stage in the development of that VE. For the RadLab VE, none of the resource acquisition took place at this stage. The radiation theory textbooks needed had already been acquired for the concept design and storyboarding stages. The other resources were not specified until later in the VE design process. It would therefore seem logical to move this activity to a more appropriate point in the VEDS.

Virtual task analysis

The design of this VE incorporated many different types of activities for the user. These included navigation, picking and placing objects, opening doors and drawers, and pressing buttons and switches. Many of the tasks in the VE incorporated a combination of these activities. The virtual task analysis was used to check not only the usability, but also the consistency of the metaphors used. For example, if a mouse click on a radiation source produced the prompt "Do you want to select this radiation source?", then a mouse click on a shield material should produce a similar prompt. The aim was also to reduce the complexity of the interactions required whenever possible. Table 5.3 shows a list of the generic tasks that are required in the RadLab VE.

1	Move to the laboratory bench	Use the joystick to navigate across the room
2	Pick up the worksheet	Click on the worksheet with the mouse
3	Read the worksheet	Click on the worksheet icon
4	Pick up a shield material	Click on a shield material
5	Pick up a radiation source	Click on a radiation source
6	Place the source and shield in their stands	Click on the source and shield stands
7	Switch on the geiger counter	Click on the green button on the geiger counter
8	Select a new shield material	Go to step 4
9	Select a new source material	Go to step 5
10	Zoom in	Click on the zoom in button
11	Zoom out	Click on the zoom out button

Table 5.3. The virtual task listing, showing a list of the generic actions.

5.3.3 Building

VR system configuration

The VR system configuration for this project was first considered during the preparation phase of VEDS. This process was finalised during the specification stage as part of the concept design.

VE appearance specification

As in the building with virtual Lego case study, the VE appearance specification had taken place as part of the storyboarding earlier in the VE design process. However, at this stage the VE developer has to work out how to meet the VE appearance specification; in particular, what level of detail is required to adequately represent the concepts being modelled and what resources are required in order to represent the concepts with the fidelity required for the application? Further, it is not just with respect of the appearance that these decisions needed to be made. In this VE one of the requirements specified was the use of realistic sounds, especially for the Geiger counter. These sounds would need to be sampled or synthesized and stored in a suitable format. Other resources required were image files to make textures for wall posters, work sheets, information sheets, radiation shield materials and the floor. As well as the visual appearance of objects within the VE.

the layout and appearance of hybrid display elements, such as the on-screen buttons and text boxes, also had to be established.

Cues and feedback specification

The specification for the design of the VE with respect to cues and feedback had been established during the storyboarding and virtual task analysis phases. At this stage more detailed design work was required on how to build and program into the VE the interactions specified. This had to be done in the context of building a VE for use in school by students, some of whom may have little experience of using computer software interfaces.

VE building and testing

Structurally, in terms of VE geometry, this was a relatively simple VE, largely because, whilst the environment had to resemble a typical school science laboratory, it did not have to be a model of any actual real world laboratory. This meant that complex or difficult to model shapes could mostly be avoided. It is normally best for the VE developer to start by building the objects that are anticipated to bring the most problems in creating a geometric structure that renders correctly. In this case study, objects with potential rendering problems could be designed out of the VE. This meant that there was freedom to develop the VE geometry in any order chosen by the VE builder, and so a conventional (in the real world) building schedule was adopted, with the room being built first, followed by the furniture and then the science equipment.

The room had to be captive (i.e. no way for the user to leave the room) to keep the users on task, and have plenty of space for easy navigation. The furniture consisted of desks, on which the equipment was laid out, and drawers, shelves and cupboards, on and in which worksheets and information sheets could be found. The science equipment consisted of a Geiger counter (actually made up of a Geiger-Muller tube and a ratemeter), the radiation source and shield materials and their stands.

All the equipment was labelled clearly and clicking on the labels for the components of the Geiger counter brought up a dialogue box with a brief description.

A simple screen layout was adopted, with the window into the virtual world taking up most of the space except for a margin to the left of the screen in which all the buttons and icons would appear (figure 5.8). The requirement for the application to be compatible with an 800x600 screen resolution limited the amount of room available on the screen, so the buttons and icons had to be efficient in their use of space.

Figure 5.8. The opening view of the 'teaching radiation' VE (version 1) showing the margin to the left for icons and buttons.

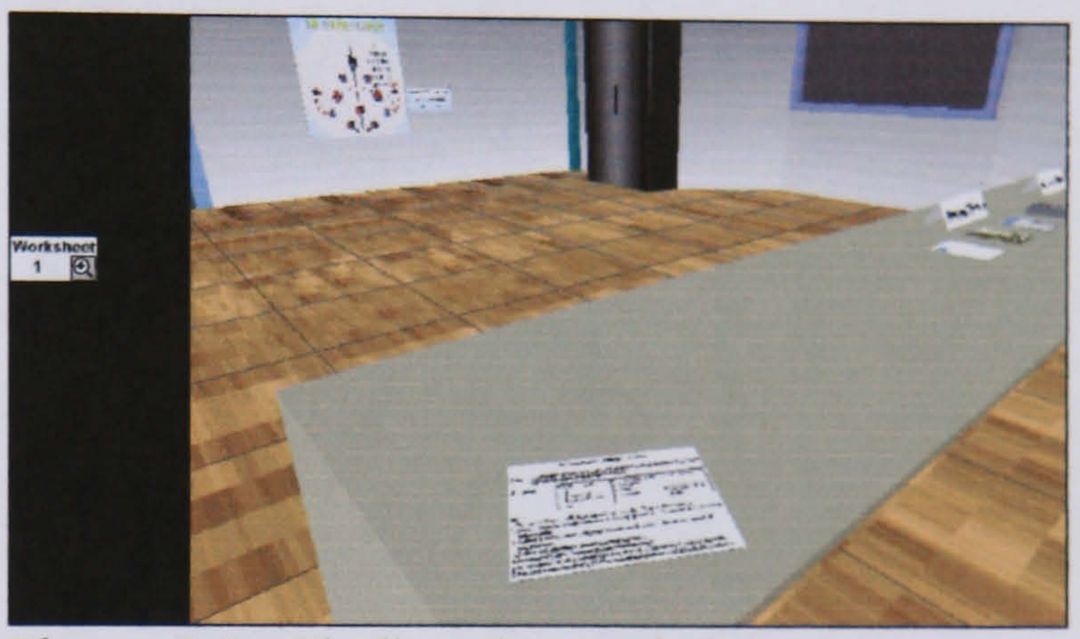


Figure 5.9. Finding the worksheet.

On entering the VE the user had to first move over to the desk where they would find a worksheet. This could be picked up by clicking on it with the mouse, at which point it would appear as an icon in the left margin (figure 5.9). Clicking on this icon would display the worksheet (figure 5.10) in a two-part dialogue box that could be closed by clicking on the OK button on the second part. This worksheet could be accessed again at any time by clicking on the icon. Following the instructions on

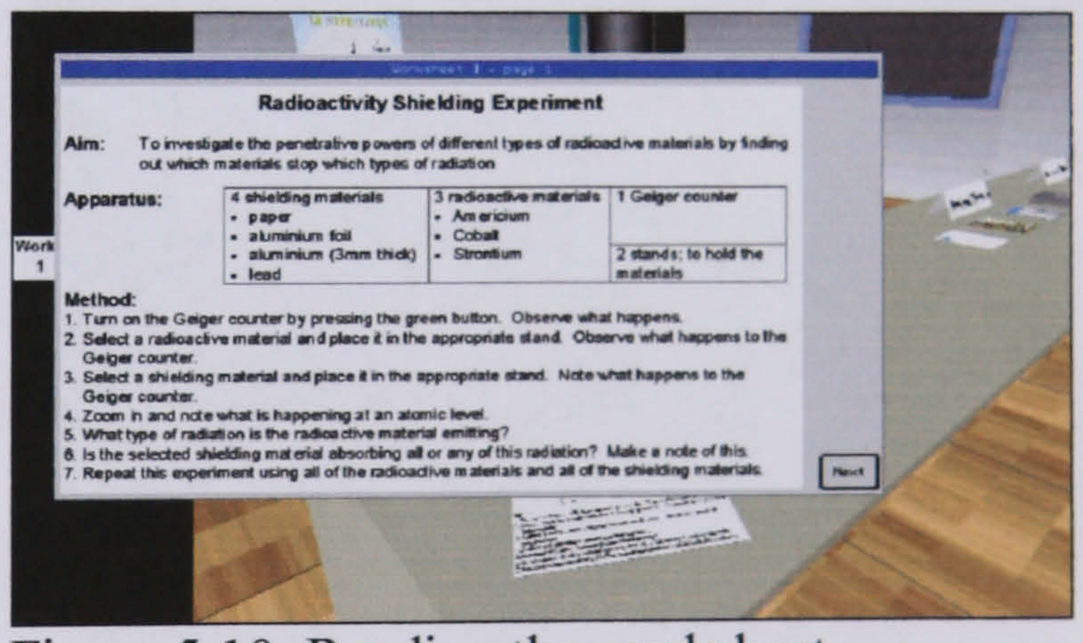


Figure 5.10. Reading the worksheet.

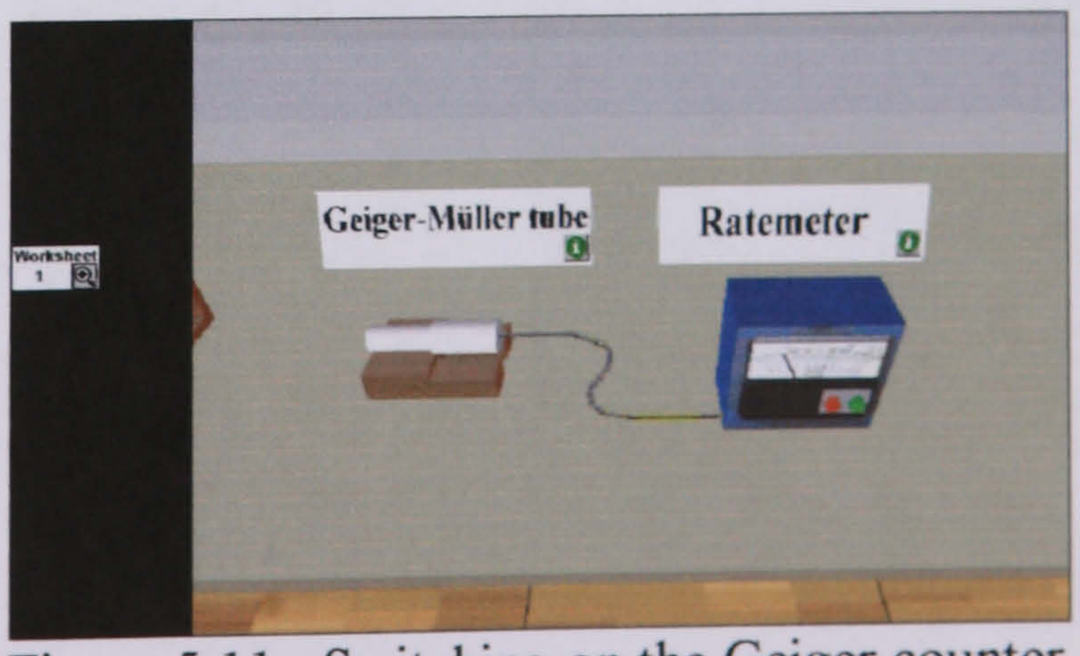


Figure 5.11. Switching on the Geiger counter.

the worksheet the user was asked to switch on the Geiger counter by clicking on the green button (see figure 5.11). This would cause the Geiger counter needle to move and a clicking sound to be emitted as if it was detecting background radiation. The next instruction was to select a radiation source. These were found on the adjacent bench along with the radiation shield materials (figure 5.12). The source was selected by clicking on one of the three labelled boxes containing the radiation sources. The viewpoint then changed to one zoomed in towards the box, which would then open revealing the source, and a dialogue box would appear in the margin asking the user "would you like to select this source material? YES NO" (figure 5.13). Clicking on "YES" selected the material which then appeared, held by tweezers, as an icon in the margin (figure 5.14). The viewpoint zoomed out from the box at this point. Next, the user had to select a shield material. The method of doing this was the same as for selecting the radioactive source, except that the shields are not stored in boxes (figure 5.13). With both a selected source and shield displayed in the left

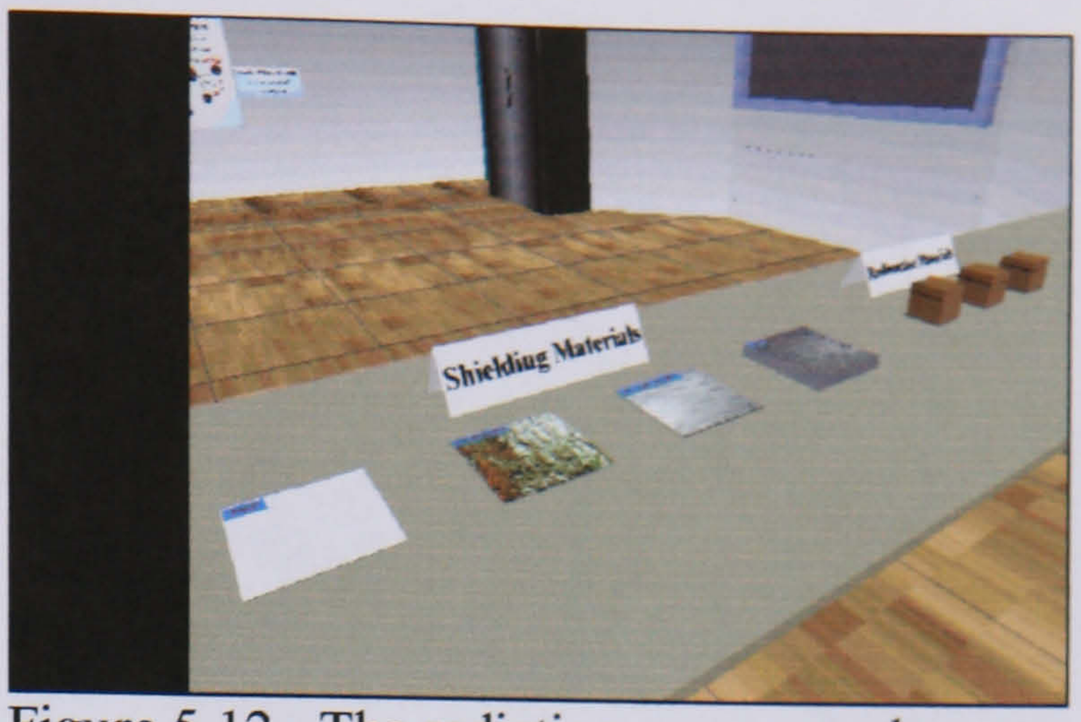


Figure 5.12. The radiation sources and shields laid out on the bench.

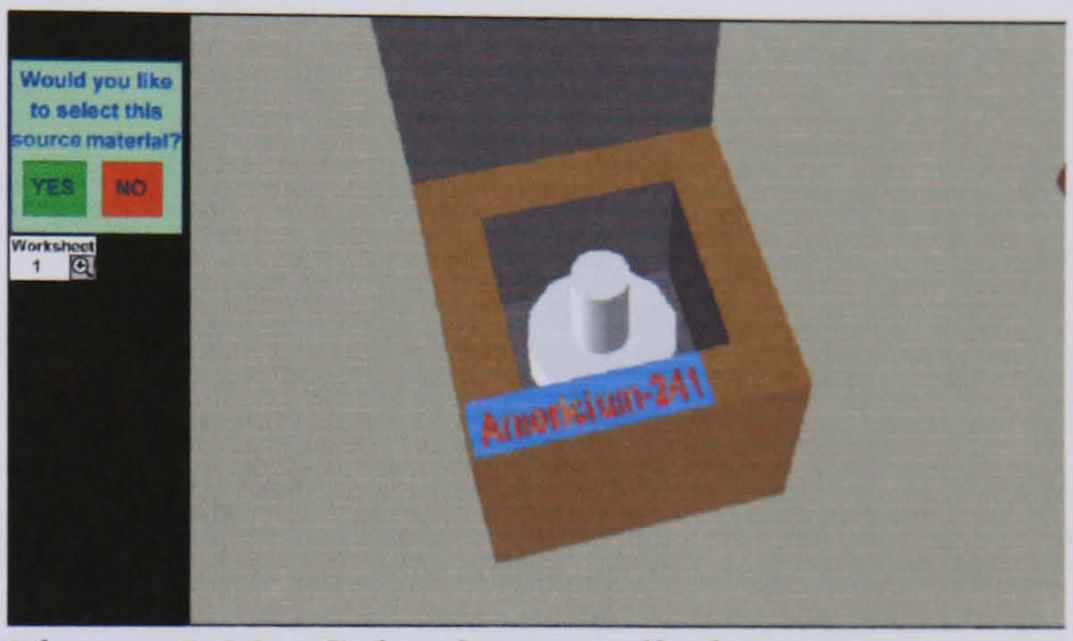


Figure 5.13. Selecting a radiation source.

Figure 5.14. Selecting a shield material, the selected source is shown in the margin.

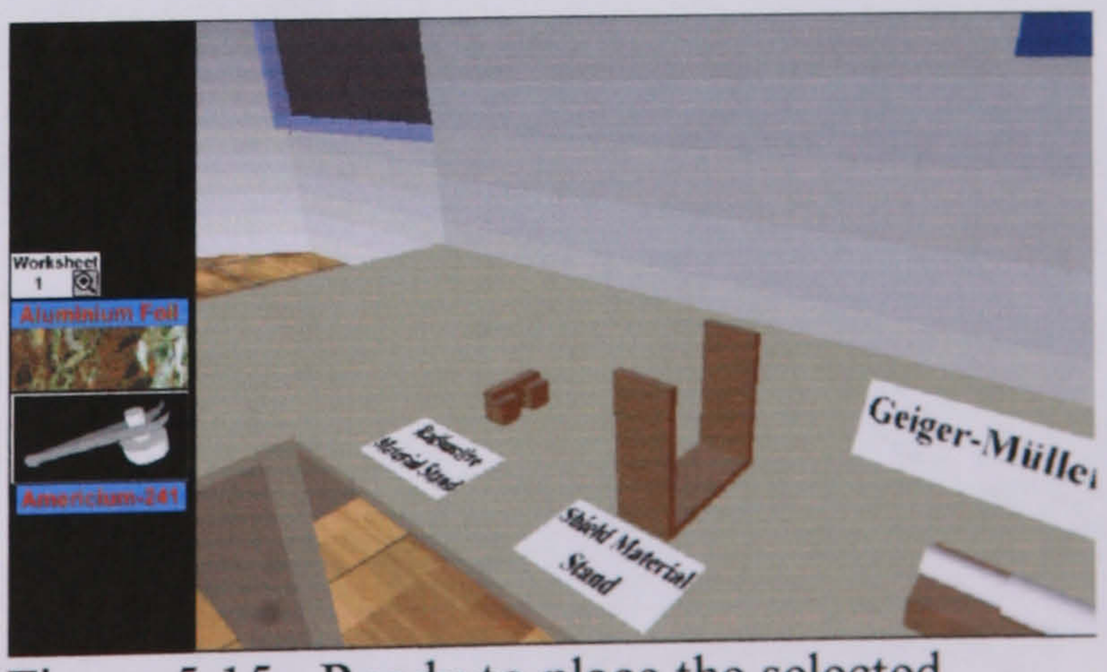


Figure 5.15. Ready to place the selected source and shield into the experiment

margin, the user was then ready to put them onto the stands provided for the experiment (figure 5.15). This was done by clicking on each of the stands, the shield and source disappearing from the left margin and appearing on the stands (figures 5.16, 5.17). As these elements were added to the experiment the reading on the Geiger counter dial (figure 5.18), and the frequency of the emanating clicking sound changed according to what source and shield material were selected. The user could then record this reading. Other activities available included looking at the posters on the walls of the room (figure 5.19), and finding and reading the information sheets in the drawers (figures 5.20, 5.21).

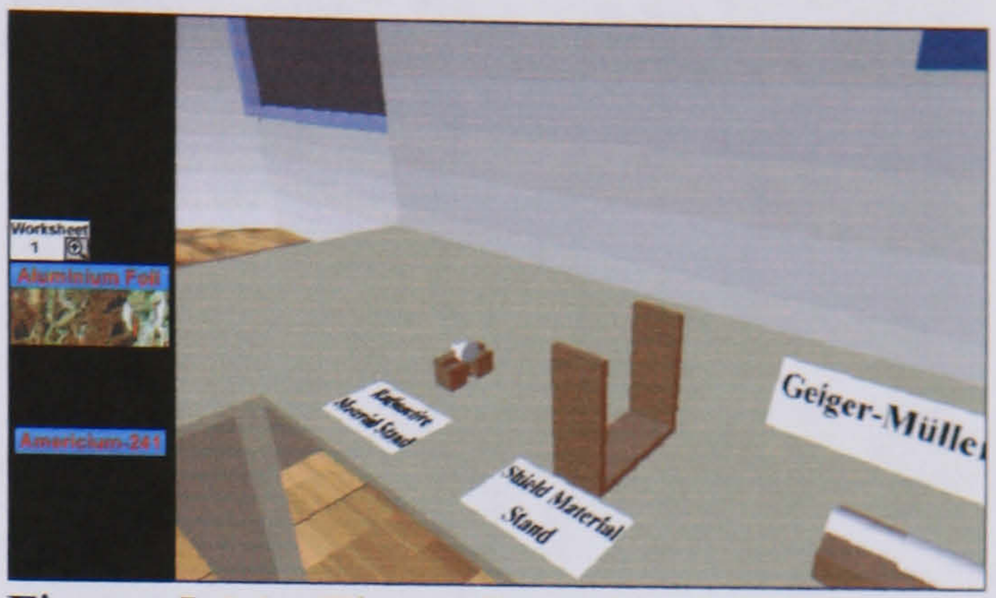


Figure 5.16. The radioactive source is placed on its stand.

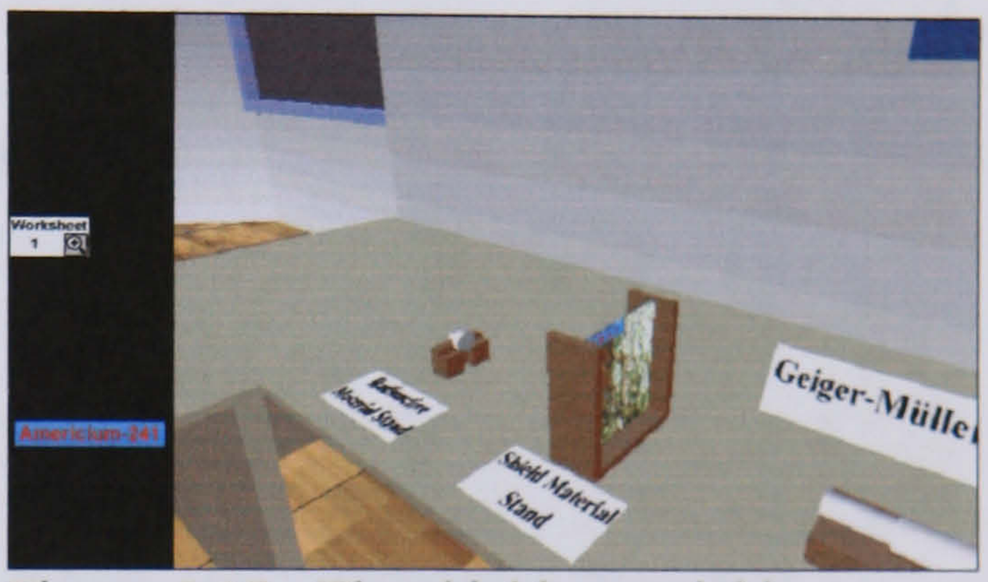


Figure 5.17. The shield material is placed on its stand

After the initial specification and design stages the development process changed to an iterative one of testing, redesign, rebuilding and retesting. Most of the testing was carried out in schools, on school computers, by science teachers, and later by school students as well. The feedback from these sessions was used to redesign the VE. The first redesign was the addition of the facility to view the experiment using a schematic representation of the atomic level. A 'zoom in' button was added to the left margin on the screen. When clicked on this changed the display to one depicting

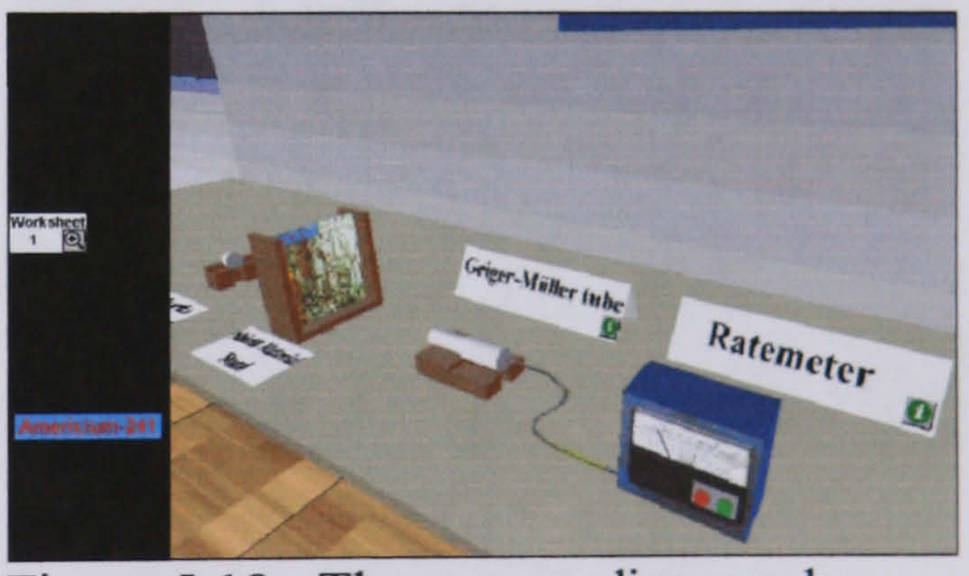


Figure 5.18. The new reading can be taken from the Geiger counter

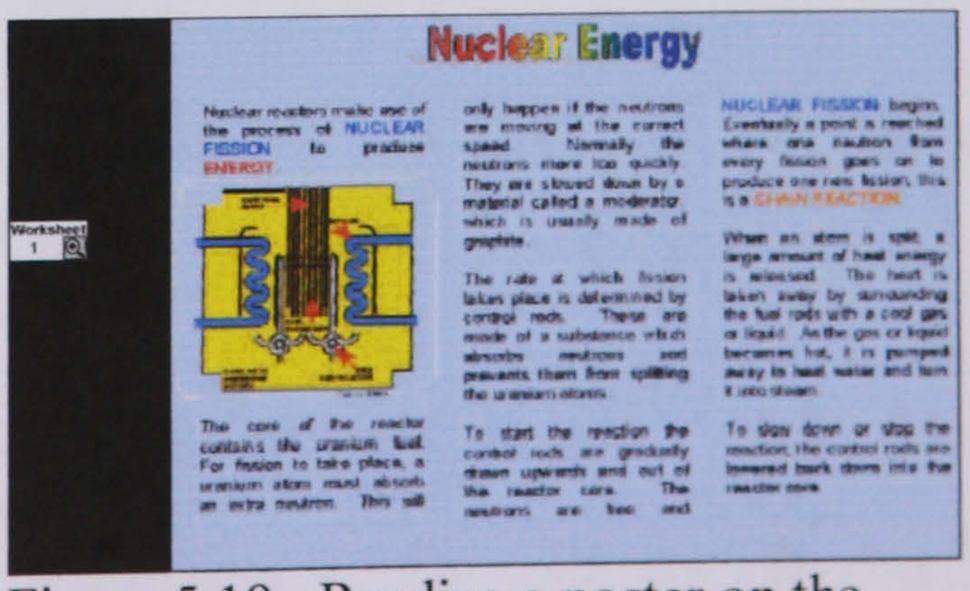


Figure 5.19. Reading a poster on the wall.

atoms, a shield and a Geiger-Muller tube (figure 5.22). Each of the atoms emitted radiation towards the tube, which may or may not be stopped by the shield depending upon the shield material and the radiation type. The type of radiation and shield material depicted related directly to the current sources and shields selected for the experiment. A zoom out button returned the user to the normal viewpoint.

The next addition was introduction screens and a help facility. The introduction screens came up automatically when the application was loaded or reset. These screens displayed the title and instructions on how to use the VE (figure 5.23). The help facility was implemented using an extra button in the margin to the left of the display. Clicking on this button brings up help text in a window that can be closed again by clicking on 'OK' (figure 5.24).

A second worksheet was added that suggested further activities. One of these activities, added at this stage, was the 'lift to the atomic level' (figures 5.25, 5.26). The teachers had suggested having adjoining rooms where the students could investigate different types of radiation. This developed into the idea of having a lift that could transport

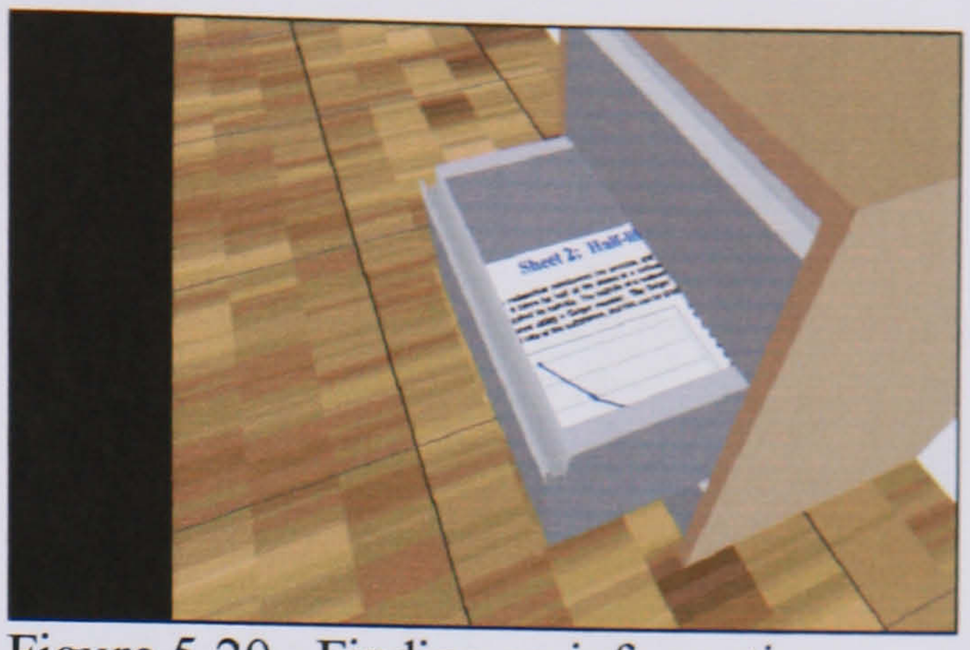


Figure 5.20. Finding an information sheet in a drawer.

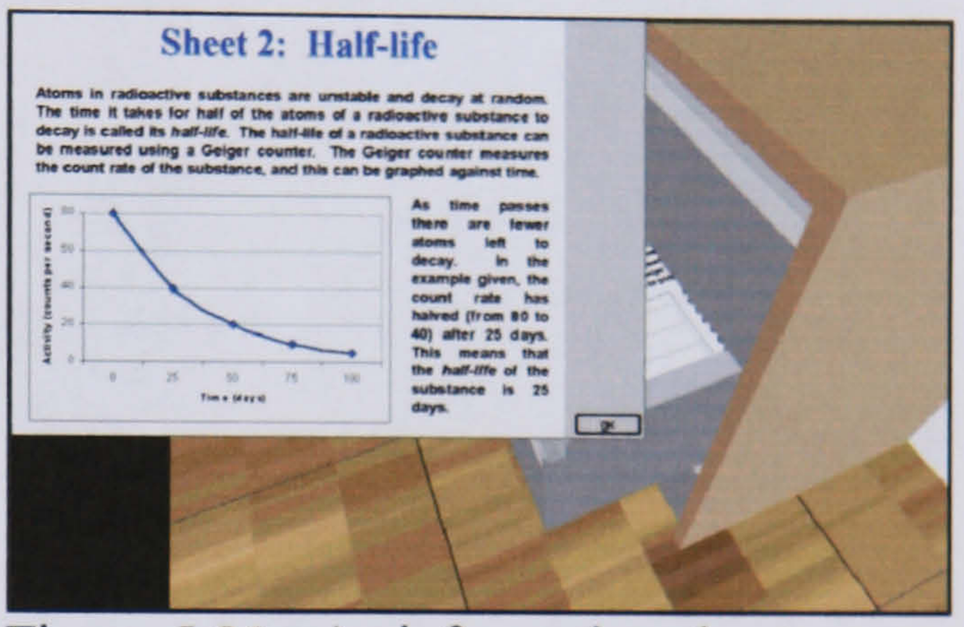


Figure 5.21. An information sheet.

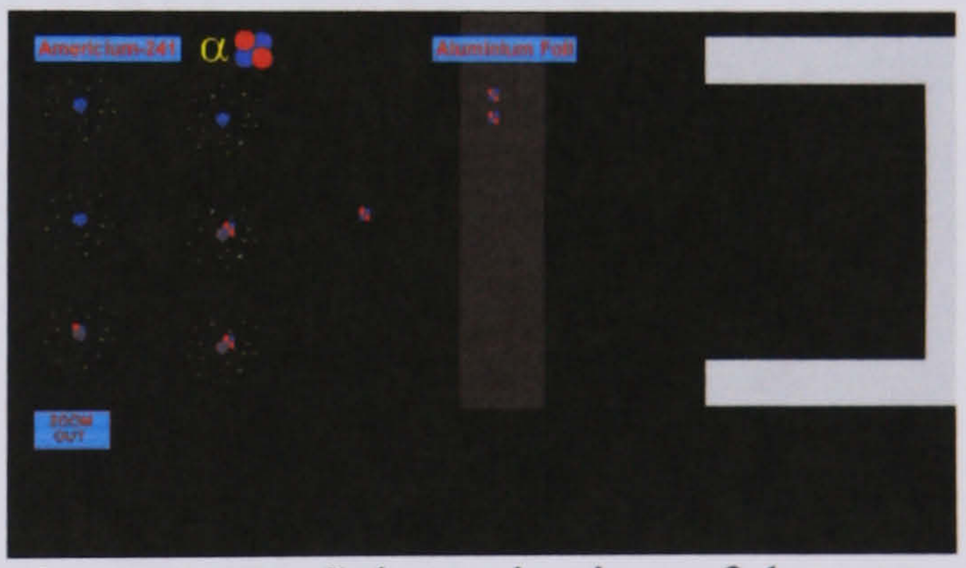


Figure 5.22. Schematic view of the experiment at the atomic level.

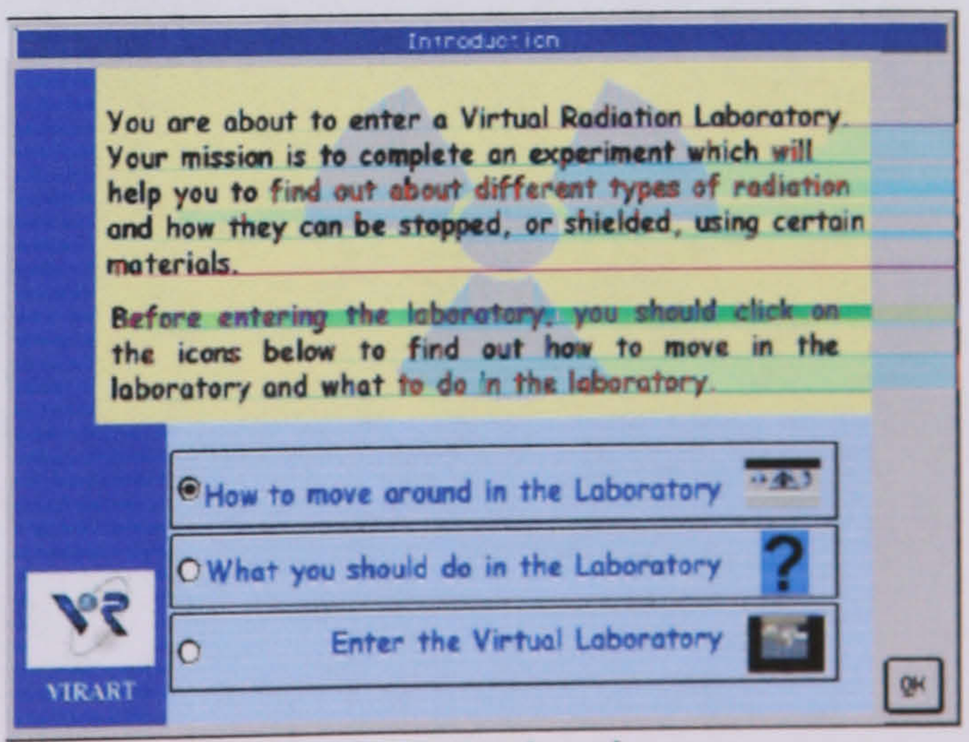


Figure 5.23. An introduction screen

the user down to the atomic level from where they could view an atom emitting radiation and read a description of that type of radiation. The lift was given four levels; the existing laboratory level, and the alpha, beta and gamma radiation levels.

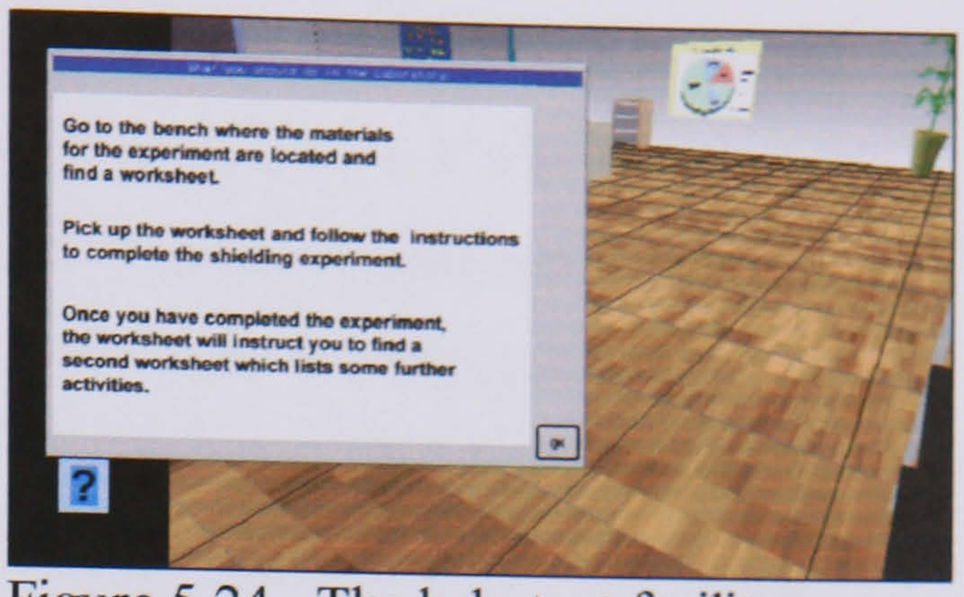


Figure 5.24. The help text facility.

The final redesign was to enable the VE to run on Microsoft's Internet Explorer (IE) version 5.0 using the Viscape plugin (figure 5.27). This would allow us to distribute the application more widely as Viscape is free to download from the Superscape website (www.superscape.com) and IE is preinstalled on most PCs when they are sold and can also be downloaded from the internet for free. Preparing the VE to be used this way involved arranging the screen layout so that it would fit within the IE window, and refining some of the images, as the Viscape format requires them to be compressed which results in losses similar to a reduction of image resolution.

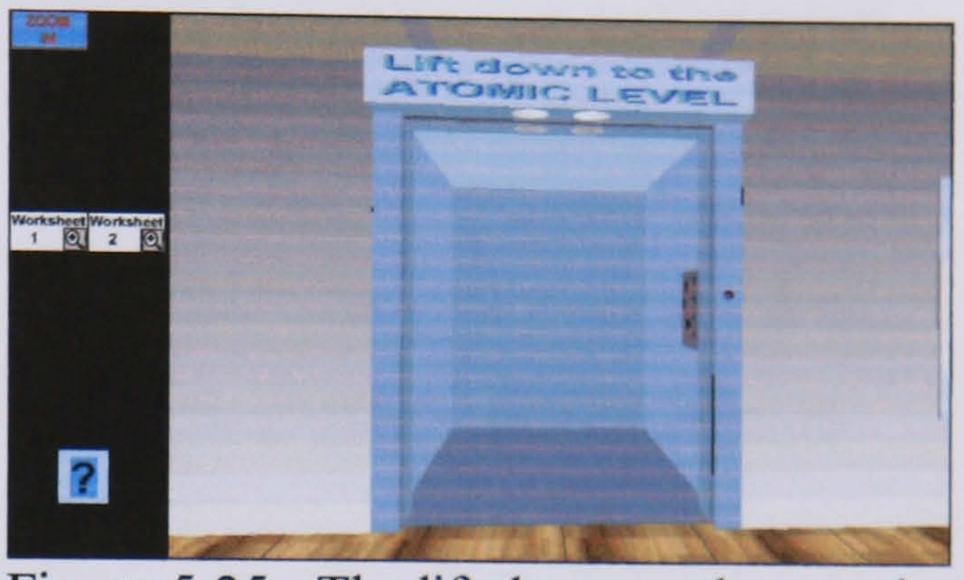


Figure 5.25. The lift down to the atomic level.

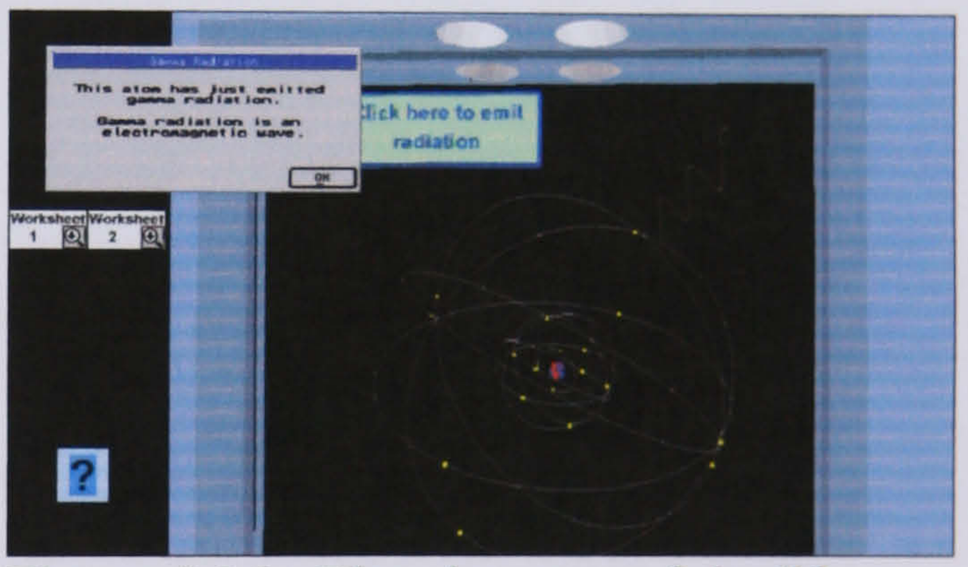


Figure 5.26. The view out of the lift at the gamma level.

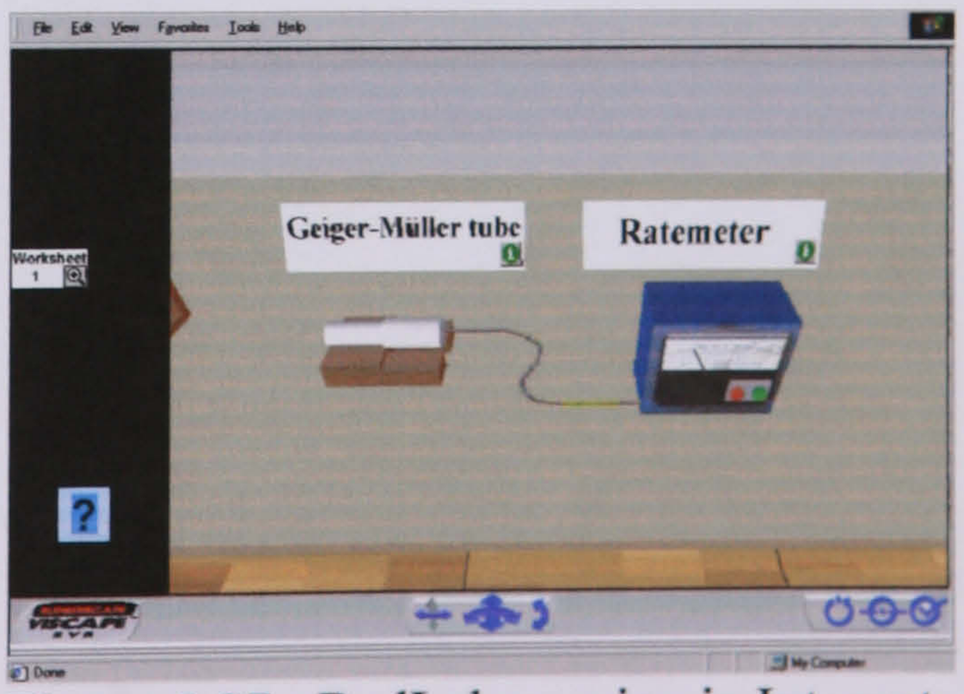


Figure 5.27. RadLab running in Internet Explorer using the Viscape plugin.

5.3.4 Evaluation

The iterative nature of the development process for this VE meant that it had been implemented in its various forms throughout the project. It also underwent three stages of classroom

evaluation and a teacher evaluation. These are all documented fully by Crosier(2000). The original aim of the project (to specify and iteratively develop a VE for teaching the specified area of study, within a user centred design process) was met successfully enough for further funding to be awarded by the Particle Physics and Astronomy Research Council (PPARC) and the University of Nottingham. This funding was provided to enable the application to be packaged and distributed to every secondary school in England, allowing further evaluation of this VE and of the attitudes of schools and teachers towards using VR in mainstream education. In summary the feedback was generally positive and informative. Negative comments were mainly based on installation and navigation issues, which could be reduced through a few changes to the VE layout and improved installation instructions for users. Respondents said that they were able to see the potential of virtual reality for teaching a range of science subjects.

5.4 Discussion

During these last two case studies VEDS was used as a guide to plan out the project work structure. However, whilst VEDS successfully defined some of the development process of these VEs, deviations had to be made from the structure described in VEDS for practical reasons that became apparent during the case studies. It was also shown that the VE specification, building and implementation stages need to be expanded and broken into sub-sections, in order to define the process in more detail. The following is a description of the ways in which the process used in the last two case studies varied from or expanded on that described in VEDS.

- The VR system configuration is usually decided upon during the concept design stage as this will influence the storyboarding of the VE.
- After storyboarding, the VE developer sets about designing the application as it is going to be built in VR. This is done with reference to the characteristics of the software and hardware of the target VR system. The process is one of working out how the concepts established in the storyboard can be realised using the tools available.
- This design process can be split into two parts, the first being an overall design phase where the sequence of building is decided and the global decisions are made, such as where objects will be located within the hierarchical geometry structures. The second phase is one of detail design and will be described later.
- Some early resource acquisition may take place prior to, or during the concept design and storyboarding stage. These resources are likely to include background information and examples of concepts to be modelled. The bulk of the resource acquisition will usually take place after the overall design of the VE has been decided upon, as it is the overall design that specifies the resources required (the storyboard may highlight a need for a resource but is unlikely to specify that need in detail). The results of the resource acquisition process will then be fed into the second, more detailed phase of the design process.
- At the core of VE development is an iterative process of detail design, followed by VE building, followed by testing.

- The detail design process consists of designing the appearance and behaviours of each of the objects in the VE, as well as the user interface and the specifics of the navigation methods to be made available.
- The VE building phase is made up of 3D modelling and world assembly. During world assembly some objects may have behaviours, sounds or textures applied to them. As the scenarios are developed, hybrid display elements such as text boxes and buttons may be added, and relevant viewpoints set up.
- VE testing can be split into three components; fidelity/validity, navigation and interaction. The purpose of this testing is to find out whether the VE developed meets the criteria laid out in the specification. If problems or shortcomings are discovered then part of the VE development process needs to be repeated. Most often this will require changes to the detail design. Other problems may lead to changes in the overall design in order for the VE to meet the existing specification. Major problems or omissions may require the re-specification of the VE. The VE will then need to be adapted accordingly, and tested again.

5.5 Conclusions

In this chapter it was shown through the practical application of VEDS, that the existing version of the structure does not adequately explain the VE design, building and testing parts of the process. From the discussion at the end of this chapter, the description of the changes required can now be used to make the required improvements to VEDS.

Chapter 6 Discussion and Development of VE design Guidance

6.1 Introduction

Throughout this thesis there are several sections devoted to the discussion of the issues arising from the case studies, and the conceptual models and techniques presented in response to these issues. The issues of navigation, interaction, and object number reduction are discussed in section 3.5. Section 5.4 discusses the Virtual Environment Development Structure (VEDS) and shows how its practical application revealed shortcoming in the areas of VE design, building and implementation. This chapter will pull together, summarise and supplement these discussions as appropriate, firstly by looking at the case studies, and then by looking at the conceptual models developed. The outcomes of this review are then used to develop a revised version of VEDS and to propose some exemplar VE development guidance tools.

6.2 The case studies

The first objective of this thesis is to use a case study based approach to establish the issues that are most relevant to the VE developer. This section discusses how this objective was met.

Case studies one to four saw the development of successful techniques to meet new challenges as they arose. Evaluations showed up the shortcomings of the spaceball and spacemouse as navigation devices for naive users (see section 3.5.3). However, matching the user to appropriate input devices is likely to remain a problem as, 'ease of use' and 'flexibility in use' of VR input devices tends to be inversely proportional. On the positive side, the placing of the salient objects of a VE within the appropriate context was shown to aid navigation and improve the users' experience within the VE (see sections 3.3.3, 3.4.2 and 3.5.2). This led to further work to find out whether existing human performance theory could be used to help VE developers with decisions about what objects to include in a VE (see section 3.6.3). From this work a list was drawn up of the types of objects that are necessary in a VE. This list is used later in this chapter in the development of an exemplar guidance tool for object number reduction and level of detail control.

In case study three, the development of a VE in which the only cues included were those that would be found in the real world showed up the shortcomings of this approach and led to significant follow-on research. The result was the development of a technique using decision tables to establish whether the points of interaction in a VE afforded sufficient cues and feedback (see section 3.6.2). One initial outcome from this was the increased provision of feedback in the form of sound in subsequent VEs. A more complete solution for desktop VR systems came with the use of hybrid displays in which the view of the VE was supplemented with a 2D interface incorporating cues and feedback not provided by the naturalistic 3D environment. Using this technique, activities too complicated to execute within a VE using the available hardware user interface could be replaced by interaction metaphors. Using metaphors, complex operations requiring (in the real world) sophisticated motor skills, can be replaced by much simpler operations using the available input devices (e.g. a mouse click). The use of this technique will inevitably lead to a reduction in the naturalism of the application as a whole, but this effect can be reduced by putting all the non-naturalistic elements onto the 2D part of the hybrid display, thus maintaining the naturalism of the 3D part of the display where possible.

Objective three of this thesis was to review the existing Virtual Environment Development Structure (VEDS) through its application during VE development case studies. This was done during case studies nine and ten, where the actual process used to develop two VEs was measured against the process as defined by VEDS. The differences were documented and are discussed in more detail in section 5.4 and later in this chapter in section 6.4. As well as being a practical application of VEDS, the final two case studies saw the application of many of the successful techniques from the earlier case studies, especially with respect to hybrid interfaces and interaction metaphors, constructing geometries and programming functionality. The final case study incorporated a user centred design approach that rigorously tested the VE's usability and which showed up, amongst other things, the need for interaction metaphors to be consistent (where possible) as well as usable.

6.3 The conceptualisation of the VE development process

Objective two of this thesis is to use the methods and techniques learned from these case studies to define models for the various aspects of VE development, such that the process can be more easily understood and thus improved. This section summarises how this was achieved.

Of the six conceptual models presented in chapter two, one shows the way in which the user interacts with the VR system as a whole, another looks at the way a user experiences a VE, and four of the models show what components are required to make a VE. One of the models (figure 2.3) begins to define the development process by describing the way that virtual objects are created, but it does not offer any guidance as such. However, these models do provide an increased understanding of, for example, the hierarchical nature of the topological structure of a VE, or how attaching a viewpoint to an object affects navigation in a VE. This systematic way of looking at the construction of VEs was then fed back into the ongoing VE development projects and used to develop more practical ways of conceptualising the process.

The projects described in chapter three led to the development of models aimed at improving VE usability. From case studies two to four, reported user problems with navigation led to the development of a model that separates the environmental components of navigation from those that relate to the user interface. From the VE developer's point of view, these aspects of navigation are dealt with separately and so any guidance provided in this area will need to take this into account. The development of the decision tables for VE interaction (see section 3.6.2 and table 3.6) provided a guidance tool that could be used pre-emptively or retrospectively. This tool and its accompanying tables (see tables 3.3, 3.4 and 3.5) showing the relationships between cues, feedback and potential for interaction, provide a useful way of looking at interaction, revealing all the ways in which interaction problems can manifest themselves as a result of the use of inappropriate or inadequate cues or feedback. However, the decision table itself is difficult to draw up and does not, on its own, review inappropriate cues from non-interactive objects (see table 3.3). Also there is no allowance in this method for taking into account the relevant attributes of the user with respect to the VE task. On the other hand, the tables showing the relationship between potential for interaction, cues and

feedback were found to be useful during later VE development projects when used proactively as a reference tool during the VE design process.

The development of the conceptual models described in this section should be viewed as part of a process rather than an end in itself. Each model contributes towards an understanding of the VE development process, but they offer neither definitive solutions, nor a framework within which to apply them. The discussion following the case studies in chapter seven showed how, with some improvement, VEDS could be used to provide this framework. Objective four of this thesis is to use the results of the review of VEDS, together with the conceptual models previously developed, as a basis for the formulation of new sections to be added to the structure for the development of VEs. The next section describes how this was done.

6.4 The enhancement of VEDS

The existing version of the Virtual Environment Development Structure (VEDS) presented in chapter six, was developed collaboratively by a number of researchers from different backgrounds (including the author). Up until this point the emphasis in this work had been to get the overall structure correct, with further work done to add detail to the sections covering evaluation. Less effort had been applied to some of the other detail especially with respect to the sections dealing with VE design and building. Chapter seven documents the ways in which the actual VE development process used in two case studies differed from that outlined in the existing VEDS, and goes on to suggest how changes could be made to improve VEDS. What follows in this section is a description of how VEDS has been enhanced to take into account the findings of chapter seven, as specified in objective four of this thesis. There are now seven sections of the revised VEDS that are most relevant to the VE developer; these are specification, VE overall design, resource acquisition, VE detail design, VE building, testing and implementation. Figure 8.1 shows how they fit in with the other, unchanged sections of VEDS, and figure 8.2 shows the reworked sections in detail.

6.4.1 Specification

This explanation of the specification process has been developed through the author's experiences gained from case studies documented in this thesis and other VE development projects. The client will have decided on the basic purpose for which the VE is being developed, but the developer may find this plan is either vague or partially impractical. The first step of VE specification is to define a list of practical, achievable goals that will give the VE the required utility. At this stage it may not be possible to establish which goals are achievable, either at all, or within the available time frame or budget. It may therefore be necessary to prioritise the goals for the developer. Other constraints on the design (such as things to be avoided) can also be raised at this stage.

Concept design is the first stage of defining how the goals will be achieved using VR technology. Background knowledge of the project domain will be required, possibly necessitating literature searches and/or some preliminary resource acquisition. Each of the key features of the proposed VE will need to be discussed with a view to deciding how that aspect of the project can be represented using VR technology. This process will

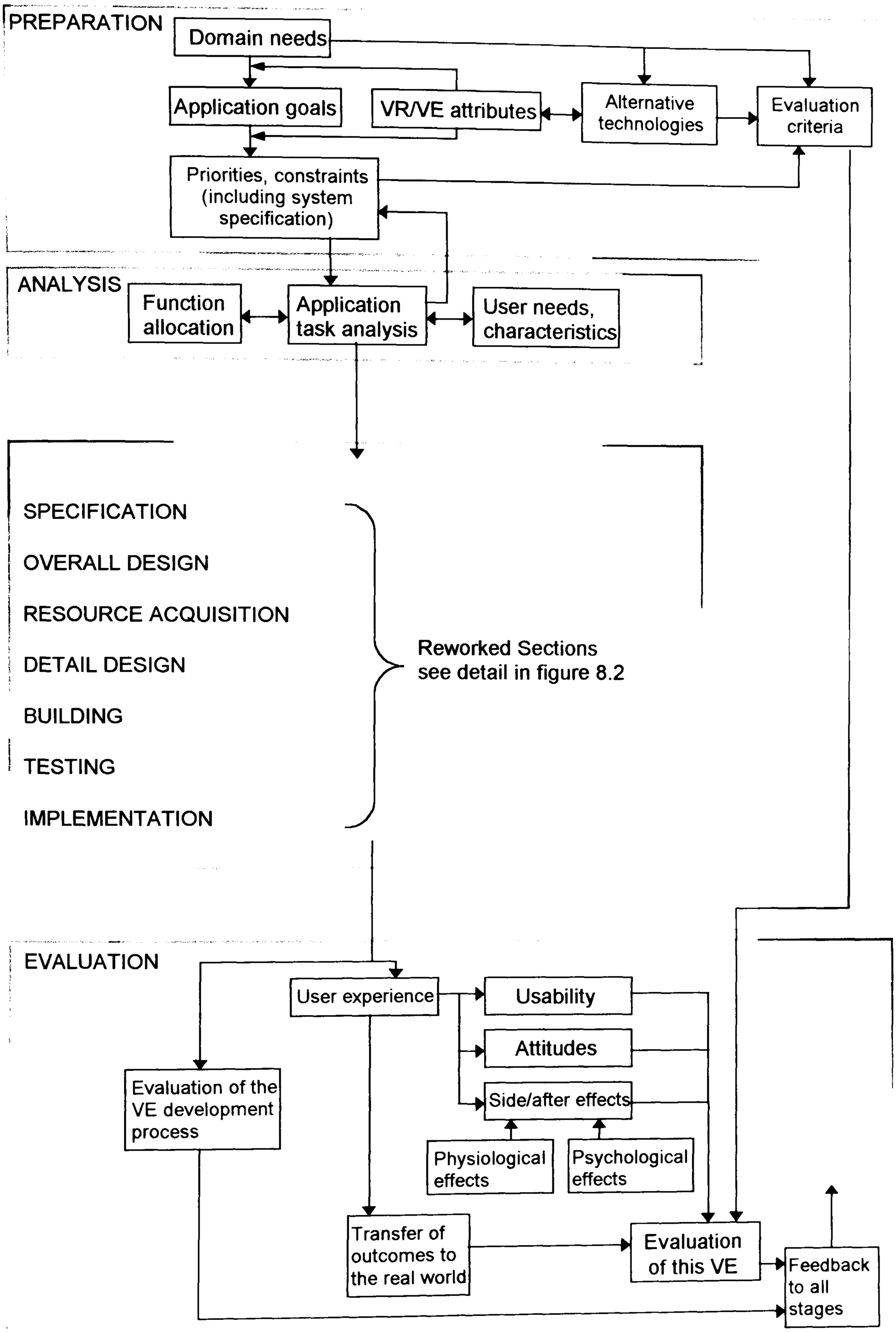


Figure 6.1. VEDS, showing where the reworked sections fit in.

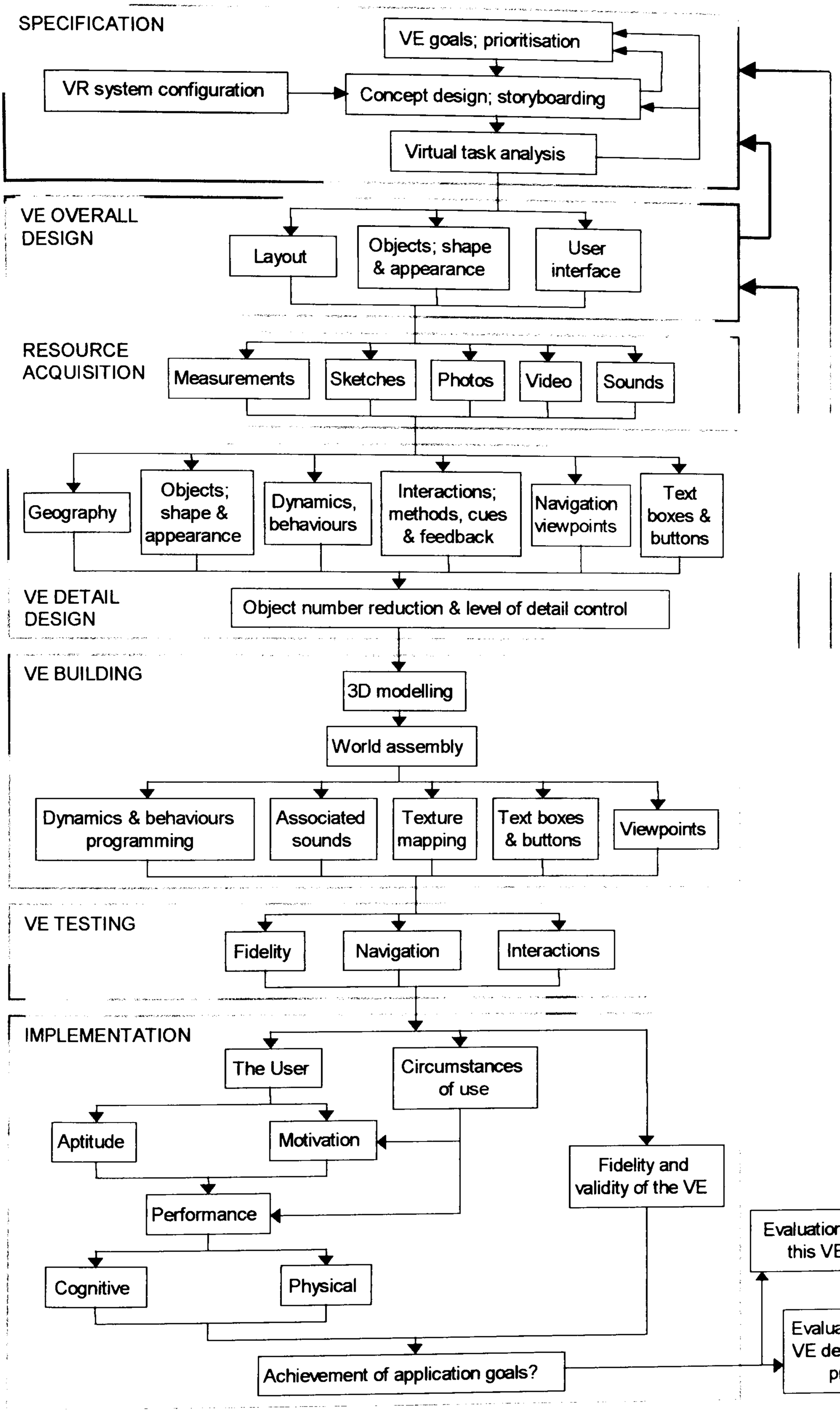


Figure 6.2. VEDS: detail of the reworied sections most relevant to the VE developer.

also include deciding on the target VR system configuration. If, as is often the case. the application is expected to run on existing user systems, some sort of analysis of available hardware amongst the expected user population will be required.

At this stage the key features described in the concept design will be isolated from each other. Storyboarding the proposed VE will unify the concept, giving it a start point and possibly an end point, as well as defining the process by which the user can move between the key features. A storyboard is a set of descriptions, each using pictures and/or words to explain a stage in the experience offered by a VE. It can differ from a storyboard for a film in that in use, a VE may not be a straightforward sequence, so the links between scenarios may be more complex. Alternatively it may not be necessary to show the links between the scenarios if for example the sequence of events does not affect the outcome.

VEs are often designed by more than one person. Frequently, they are designed by a group whose members work for different organisations and come from different backgrounds. Some of these people will have a good understanding of the possibilities offered by a VE. Others will be experts in their own field, and have a good idea of what they want the VE to do, but little idea of what is feasible. A storyboard is a method by which the different parties involved can describe, design, and agree on the form the VE should take. The developer can use the story boarding process to work out how to match the requirements of the client to the governing factors of the VR system. The storyboard can then be used by the developer to build the VE, which as a result, should closely match the requirements of the client.

It is important to get all the relevant people involved at the story boarding stage. If this is not done a VE may be developed which does not meet the requirements of the person who was not consulted. This could result in expensive and time consuming re-development. Generally the story boarding process should involve;

1. the project manager - to oversee the process, and keep it within budget
2. the VE developer(s) - to establish exactly what the client wants, and to explain what is feasible and how long it will take
3. the client (the person/people who specified the VE) - to make sure the VE will fulfil the specified requirements

4. a good representative of the user population, or a person with a good understanding of the user population - to make sure the VE will be suited to the user population.

It is often the case that one person may fill more than one of the roles described above, in which case the size of the storyboarding group will be reduced. Typically between two and five people will develop a storyboard.

What is included in the storyboard will depend on the specifics of the VE, however all features of the VE which are critical to its ability to meet the specification should be included in the storyboard. Typically a storyboard will include;

1. The initial scene presented when the user first enters the environment
2. The expected layout of the scene at subsequent points
3. Any narration or instructions, whether verbal or textual
4. Any signs or cues given to the user
5. Images to describe the activities which the user is expected to take part in
6. The method, in terms of user input device, of performing these activities
7. Images to show the consequences of various actions (both correct and incorrect) by the user
8. The sequence of events, or where there is no sequence required (i.e. the user can perform the tasks in any order) this should be stated
9. Any links with other software, e.g. multimedia clips
10. If the experience has a formal 'ending' this should be included

On top of this it may be necessary to prioritise the contents of the VE, especially where time constraints on the developer will mean that some elements may have to be omitted. Prioritisation also helps where a complex VE results in a choice between levels of detail and rendering speed, or between conflicting interaction metaphors.

Once the developer starts building the VE they will be confronted with a lot of unforeseen decisions at every stage. It may not be possible for the developer to make some of these decisions and they will have to be referred back to the relevant person in the storyboarding group. This could delay the development process if the lines of communication are not good. It is therefore important to establish as much detail as possible at the storyboarding stage. The developer has to drive this process by asking questions about the design of the VE. Even if the answer is "it doesn't matter (about that detail)" it is necessary to establish that that particular decision is left to the developer.

A virtual task analysis is an analysis of the tasks to be performed by the user within the VE (as against an analysis of the tasks as they would be performed in the real world). There are three main purposes to this analysis; the first is to establish whether the user will find it easy to carry out activities in the VE. For example whether additional cues or feedback will be needed to guide the user through the tasks, or whether the virtual tasks are easy to carry out with the proposed input devices. The second is to find out whether the activities as modelled in the VE are conducive towards achieving the goals of the VE, for example in achieving learning objectives (e.g. (Neale et al., 1999)). The third purpose is to reveal any potential health and safety related side effects that may occur as a result of using the VE.

The entire specification process is an iterative one, with the results of the virtual task analysis feeding back into the storyboarding possibly resulting in modifications to improve the proposed VE's usability or utility. Both the virtual task analysis and the storyboarding can result in issues being raised that lead to redefinition or reprioritisation of the VE goals. Further, the specification process may be revisited later in the light of issues thrown up mainly by the VE design and VE testing phases of the process. It is during the VE design phases that omissions or oversights from the specification phase will be noticed. The VE testing phase will show up obvious flaws in the VE specification and design.

6.4.2 VE overall design

After specification, the next phase is the design of the application as it is going to be built in VR. This is done with reference to the characteristics of the software and hardware of the target VR system. The process is one of working out how the concepts established in the storyboard can be realised using the tools available. During this phase the VE developer will need to plan out the development process so that the finished VE can be delivered in the time available, the resources needed can be prepared, and so that the different parts of the VE and the user interface will work together when assembled.

When planning the layout of a VE the first thing to think about is the extent of the model. Whether the model is on the micro (e.g. sub-atomic) or macro (e.g. an entire city) scale, there will be a system boundary, beyond which no more features will be modelled. There

may also be a second boundary within this, beyond which the user is not able to navigate. The purpose of this second boundary is to prevent the user navigating to a point where they can see outside the modelled area. The next decision for the VE developer is which way to orient the world with respect to the VR system axes alignment. In a VE consisting largely of orthogonal cuboids this choice will be an easy one. In a less structured VE, making a good decision at this point could prevent rendering or dynamics problems later on. Inextricably linked to orientation (in Superscape VRT at least) is the planning of the hierarchical structure of the objects. Grouping the objects into hierarchies speeds up the rendering process and allows multiple objects to be manipulated easily, however the groups are also orthogonal cuboids. So the VE layout has to be designed such that as many of the objects and groups as possible, but especially the more complex or important (for the purpose of the VE) objects, are oriented orthogonally.

The objects which are to be included in the VE need to be prioritised with respect to the overall purpose of the VE, and the resource requirements (dimensions, textures, etc) need to be specified.

The final area for planning the overall design of the VE is the user interface. The VE developer needs to decide at this stage what viewpoints are going to be available, what the screen layout will look like, and what methods of interaction will be possible. This will prevent inconsistencies in the user interface or unnecessary duplication of points of interaction (e.g. an overlaid button with the same function as an interaction point within the VE).

6.4.3 Resource acquisition

During the VE overall design phase the resource requirements for the building of the VE should have been specified. The types of resources required in order to model an object depend on what role that object will play in the finished VE. With reference to resource acquisition, objects can be split into four types. (1) Background objects are there to act as scenery at the periphery of the modelled VE. (2) Context objects are not individually fundamental to the VE, but collectively help the user to establish what type of environment they are in. Primary features are fundamental to the VE, but (3) visual primary features only need to look correct, whereas (4) functional primary features may need to be both functionally and visually correct.

A background object has to look right from a distance. This can normally be achieved using a texture applied to a large 2D object that cannot be viewed close up or from an acute angle by the user. A photograph of the real location being modelled or a typical location of the right type can be used to create the texture.

A context object does not need to accurately represent an actual object of that type, but rather it must be recognisable for what it is and should be accurate enough not to look wrong. As many context objects exist at the edges of areas navigable by the user these can be replaced by 2D textures where this is easier than modelling the object in 3D.

The accuracy required when modelling primary features of a VE will depend on the exact role that the feature plays in the VE, but they would normally need to be visually accurate. Where the VE is representing objects that exist in the real world this would require dimensions measured from the real object being used, along with textures to add realism where necessary. If a primary feature is functional this can increase the dimensional accuracy required (i.e. so that the object fits accurately into the same space as it would occupy in the real world). Table 6.1 shows the accuracy requirements during resource acquisition for various object types.

<table>
<tr><th>Model accuracy >

Type of object \/</th><th>Approximate</th><th>Visually accurate</th><th>Mechanically accurate</th></tr>
<tr><td>background</td><td colspan="2">2D texture</td><td></td></tr>
<tr><td>context</td><td colspan="2">2D texture(s) AND/OR 3D dimensions</td><td></td></tr>
<tr><td>primary feature - visual</td><td></td><td>3D dimensions + textures</td><td></td></tr>
<tr><td>primary feature - functional</td><td></td><td colspan="2">3D dimensions + textures AND/OR 3D dimensions (+sound? +video?)</td></tr>
</table>

Table 6.1. Types and accuracy of resource information required when acquiring resources for different object types

As well as dimensional measurements and photographs for the creation of textures, it may be necessary to collect other forms of information about the objects to be modelled.

Sketches of the objects and their relative positions (with dimensions added where necessary) will be easier to understand than written descriptions. A video of a place or a piece of equipment in action will store a wealth of information that can be referred to repeatedly as necessary. Functional objects may have sounds associated with their function that will need to be recorded (or extracted from video footage) so that they can be added to the VE.

It is common for the location at which the acquisition of the resources takes place to be different from that in which the VE development takes place. In these circumstances it is important to maximise the amount acquired in each visit, as it is better to have extra redundant information than not enough information once VE building starts.

6.4.4 VE detail design, VE building and VE testing

The detail design, building and testing phases of VE development are so closely interwoven that it is sometimes hard to distinguish between them. It is common for developers to work on one object at a time, designing, building, importing it into the VE, adding behaviours and interactivity where required, testing and refining it before moving on to the next object. Where objects are interdependent, several objects may be worked on at once. Thus, this period of the development of a VE is a repeating pattern of detail design, building and testing, at the end of which, an initial version of the finished VE will have been created. Occasionally during this process, omissions or flaws will be found that require a brief return to the specification or overall design phases. The order in which objects are designed, built and tested for the VE will be determined by the prioritisations defined in the specification, the nature of the VE itself, and the preferences of the VE developer. As well as virtual object creation the developer will also be setting up viewpoints that afford appropriate navigation, and any other user interface features such as text boxes and buttons overlaid on the screen. This user interface programming will be integral to the process of object creation; as new objects are created, changes to the methods of navigation available, buttons to control the behaviour of those objects, and text boxes giving relevant information, will be set up as necessary.

Of these three phases, the VE detail design phase is concerned with taking each detail in turn and seeing how it can best be made to contribute to the overall requirements of the

VE. These details can be; object geography, shape or appearance, dynamics or behaviours, interaction methods, or they can be the user interface issues of navigation viewpoints, on screen buttons and text boxes. It is during this detail design phase that the developer is faced with a huge number of decisions. Each decision has to be made with reference to the overall goal of the VE, and the specific requirements of the expected user population. Having come up with a design it must be considered in terms of its impact on the rendering speed and frame update rate of the finished VE. The number of decisions required at this stage in the VE development process demonstrates a need for tools to enable the VE developer to devise appropriate design solutions.

To a certain extent VE building has to be done in a fixed order. A 3D shape has to be created before it can be imported into the VE. Once it is in the VE it can be assembled with other shapes, have colours and textures mapped onto it, and be given dynamics and behaviours, and associated sounds where appropriate. Viewpoints, on screen buttons and text boxes can be added at any time.

Each detail added, and the VE as it is assembled, should be tested against the criteria set out during the specification stage. The VE developer, or an available colleague, will initially carry out this testing. It is also a good idea to periodically run tests using the client or members of the anticipated user population as subjects. The main issues to be considered are; (a) fidelity, or does the VE adequately represent the concepts as required by the specification?; (b) navigation, or will members of the anticipated user population be able to move around the VE satisfactorily?; and (c) interactions, or will members of the anticipated user population know when to, and be able to interact successfully with objects in the VE? Problems discovered at this stage could be the result of mistakes made during any of the earlier phases. Changes in the detail design will be less costly than changes to any of the other phases. If more resources are required it could involve another visit to a remote site, and if changes to the specification are needed this will involve a reconvening of the VE specification team. However it is the VE developer's job to recognise when a VE specification cannot be met and instigate the required action.

6.4.5 Implementation and evaluation

Whilst they may not be present at the time, the VE developer needs to understand the process of implementation of a VE. Once a VE has been delivered to a client there are

three factors that may determine the success of its implementation and the achievement of the application goals. Firstly there are the characteristics of the actual users, which may or may not match those of the anticipated user population, particularly in the areas of aptitude and motivation. The user's motivation may be further affected by the circumstances of use. Factors such as time of day, what they would normally be doing (instead of using a VE), as well as environmental factors such as ambient temperature and background noise, could affect the user's motivation and overall performance in the VE. Finally, the fidelity and validity of the VE itself, resulting from the earlier specification, design and building phases, will certainly have an effect.

VE evaluation requires entirely different skills to those required for VE development and is therefore likely to be carried out by different personnel. However it is only through evaluation that the true effectiveness of the VE can be established. By working closely with VE evaluation experts the VE developer can improve the VE being evaluated, and learn much about which VE development techniques result in a successful VE.

6.5 Exemplar VE development guidance tools

Objective five of this thesis is to identify parts of the VE development process where extra guidance is required in order to avoid VE design problems, and objective six of this thesis is to create exemplar VE development guidance tools to help with specific aspects of the VE development process. Rather than providing guidance that can be applied retrospectively in order to correct any design errors made during the VE development process, it is the view of the author that guidance will be more usefully applied to support the detail design phase. Section 6.4.4 identified the VE detail design phase as one where tools are required to help VE developers cope with the large number decisions they are faced with. To make it useful to a VE developer, any guidance has to be simple and quick to apply. This can be done by keeping it brief and to the point. The guidance can be backed up by further, more in depth information and references, should they be required. The VE detail design section of the VEDS (figure 6.2) shows it to be split into seven areas. These are topography, object shape and appearance, dynamics and behaviours, interactions (methods, cues and feedback), navigation and viewpoints, text boxes and buttons, and object number reduction and level of detail control. It is relatively easy to provide one-off individual pieces of advice in any of these areas for a particular application. It is much more difficult to integrate and provide this advice in a structured way and make it comprehensive enough to cover the most common requirements of the VE developer. It may be that not all of these areas can be supported by a structured and comprehensive guidance tool, but following the case studies reported in this thesis the author has attempted to provide such guidance for interactions (methods, cues and feedback), navigation and viewpoints, and object number reduction and level of detail control. The different nature of the guidance required in each area leads to them being presented in different ways. For interactions, the guidance offered is fairly precise but it depends on various factors (e.g. object appearance), so a flow chart representation seems to be appropriate. For navigation the recommendations are more complex and less easy to define so a table format is used. For object number reduction a simple checklist can be used to establish whether an object should be included in a VE. This checklist is also presented in the form of a table.

6.5.1 Interactivity design guidance tool

A major challenge to the VE developer when implementing interactivity is to appropriately represent the affordances of a virtual object or system. That is, the VR system needs to provide the user with enough information to establish what interaction is possible in the VE, and also what interactions are not available. Figure 6.3 shows the design guidance tool for interactivity. The first decision the designer is asked to make is whether the object being designed is going to be a point of interaction. When designing objects for interactivity it is important to consider the absence of, as well as the presence of, potential for interaction, in order to avoid the problems caused when an object that does not afford interaction is interpreted by the user as affording interaction (see table 3.3). This misinterpretation is most likely to be a result of the visual appearance of the object. Objects that look as though they afford interaction may have been included in the VE for reasons of fidelity, but to program the interactivity to those objects would be a waste of the developer's time if that interactivity were not required by the VE specification. For example, most rooms have a light switch. For realism that light switch could be modelled in a 'room' VE. But the ability to switch on the light in the room may not be a requirement for the VE. Because the switch is there, the user may try to switch the lights on/off, or - from table 3.5 - 'the user may try to interact with the object, but will not know whether that interaction was successful' (inappropriate cue, no immediate feedback). The guidance tool suggests three possible solutions to this. The first is to eliminate the object from the VE. This is certainly the simplest solution but it may have a detrimental effect on the VE (see section 6.3.3). The second option is to make the object look non-interactive. In the case of the light switch this could be done by modelling the mounting plate but not modelling the switch itself. A similar technique is to reduce the level of detail (LOD). This is more appropriate to objects incorporating complex geometry and/or texture mapping, where this complexity can be reduced to the point where the object is recognisable but not attractive to the user. The third option is to add feedback to the object (in fact making it interactive) that informs the user that the object does not perform any function. Once a non-interactive object has been adapted as necessary using one of these solutions the developer can move on to the next object.

If, on the other hand, the object being designed is an interactive one, the challenge is to make the object look interactive and to give some indication as to what will happen if

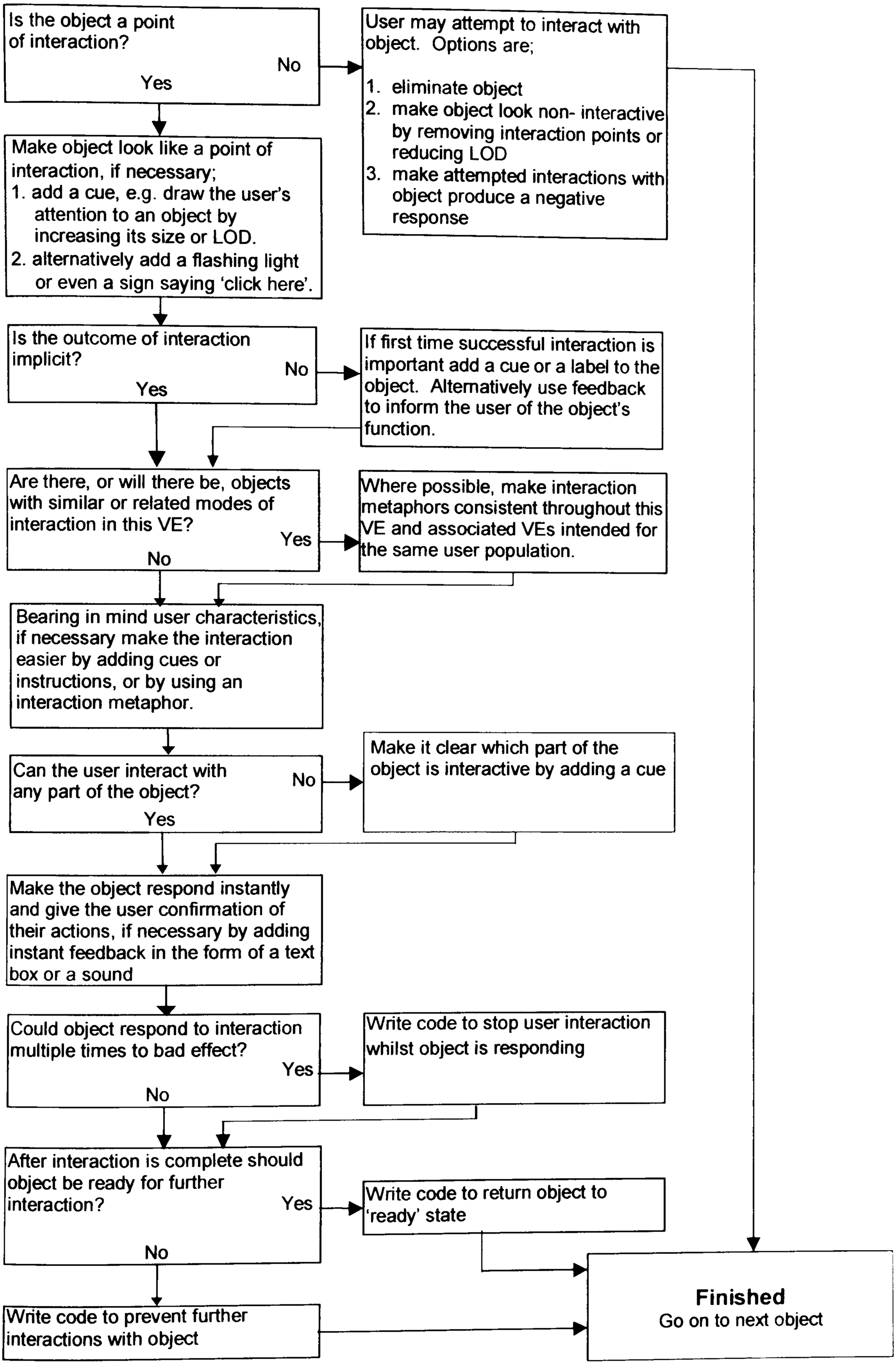


Figure 6.3. Flow chart of decisions to be made when designing interactivity into a VE.

interaction is successful. As stated previously, some objects have their interactivity implicit in their appearance. Using the light switch again as an example. most users would know that it can be switched and would expect a light to come on or go off as a result. A similar switch mounted on an unfamiliar piece of machinery might not immediately imply a specific outcome to interaction, whilst a pressure pad, for example, might not look like a point of interaction at all. When modelling these devices in a VE, a cue such as a label might therefore be used. In some circumstances adding a cue might not be appropriate (it might detract from the realism), in which case appropriate feedback could be relied on to inform the user of the result of the interaction. The object will still need to be enhanced in some way to attract the user to interact with it in the first place. This enhancement could be in the form of exaggerating its size or increasing the LOD used (relative to non-interactive objects).

The method of interaction with a virtual object will largely depend on the VR system input devices used, and the characteristics of the expected user population. Most VR systems incorporate some form of 'pointing and clicking at objects' metaphor that results in interactive objects responding accordingly. Sometimes interaction is performed by pointing, clicking and then dragging an object in a specific direction. Occasionally navigation by the user results in a response from objects (e.g. the automatic glass doors in section 3.4.2 and figure 3.5). These interaction metaphors need to be designed so that they match the abilities of the user population to the activities available in the VE. Making the interaction metaphors consistent throughout the VE (and other related VEs) where possible, avoids having to give instructions for every interaction and makes the VE more intuitive in use. Where a particular interaction requires a unique interaction metaphor this may need to be accompanied by instructions.

Often an interactive virtual object will be programmed to respond to being activated ('clicked on') anywhere on its geometry. This benefits the user, especially when one considers the problems of interacting with a small switch on a machine, using an unstable point and click device (e.g. a 3D mouse) and a low-resolution display. It is much easier for the user if the whole machine is the point of interaction. Occasionally it is not possible to make an entire object a point of interaction, for example where an object affords more than one type of interaction. In these cases it is important that the points of

interaction are made apparent where they are not implicit. This can be done with a visual cue as described earlier (e.g. increased LOD, labels etc).

Finally, there are three interaction problems that VEs have in common with conventional 2D computer interfaces. These are; providing suitable feedback, preventing unintended multiple interactions, and re-initialising a repeatable interaction or preventing the repeat of a 'one-off' interaction. Preventing unintended multiple interactions and re-initialising or preventing further interactions are a case of applying the appropriate computer code to the objects concerned (for example, by using a flag variable that has to be 'zero' for interaction to be possible. When interaction takes place this flag is set to 'one'. Further interaction can only take place when this flag has been reset to 'zero'). Feedback problems however, can be more difficult to sort out in a VE than in a 2D interface, as in some cases the system being modelled in a VE does not afford useful feedback in the equivalent real world system. As explained in section 3.6.2; to enable a user to successfully interact with a VE, immediate and appropriate feedback has to be provided to let the user know what the results of their most recent interaction are, or at least that they have initiated some sort of process. It may be appropriate in some cases where normally there would be no feedback, to add unrealistic feedback (a sound, a voice or a text box) to the virtual system in order to let the user know what they have done. For example, some electrical heating appliances do not provide any immediate feedback when they are first switched on. The real world user has to wait for the appliance to warm up for confirmation that the device is working. A virtual model of such a device should provide more positive and immediate feedback. Once the design of an interactive object has been through these processes the developer can move on to the next object.

6.5.2 Navigation guidance tool

The design of the methods and facilities for navigating in a VE is far from straight forward. As a result the design guidance provided here is not as concise as that provided for interaction. The VE developer will be required to use their judgement in designing a user interface that is likely to be usable by the anticipated user population, who may not have much experience of using 3D user interfaces. However, VE developers tend to be very good at using the navigation devices and metaphors used in VR systems. On top of this a developer may not have access to all the types of hardware (navigation and display devices) that will be used with the VE under development. They are therefore unlikely to

spot all of the potential problems that a more naïve user population may have, especially with specific hardware configurations that have not been tested. It is therefore more difficult for a VE designer to pre-empt all of the navigation problems that are likely to be encountered, and some revision of the VE is likely to be necessary. So rather than being a definitive set of procedures, this guidance tool (table 6.2) attempts to highlight the relevant criteria which must be considered when configuring the navigation facilities of a VE.

	ITEM	Guidance
1	**INTERFACE**	**CRITERIA**
1.1	**Choice of hardware input device(s)**	This may be predetermined, but if not, choose one to match the expected skills of the user population to the demands of the VE (Crosier, 1996).
1.2	**Choice of display device**	This may be predetermined. The quality of the display device may affect the ability of the user to recognise objects and understand their location.
1.2.1	Resolution	Higher resolution gives more detail, if the geometric model and/or textures can provide it, but the trade-off is a lower screen refresh rate
1.2.2	Colour depth	This is less important to navigation than resolution is, but for example, more colours are needed for smooth shading (which increases fidelity). Also a trade-off against the screen refresh rate
1.2.3	Field of view	Increased peripheral vision will help users understand the local layout (Sholl, 1993)
1.3	**Layout of the display** (if applicable)	Some display formats (e.g. flat screen) afford the use of hybrid interfaces combining a window onto the 3D virtual world with some form of 2D interface controlling, or giving information about, aspects of the VE. The layout of this display could affect navigation positively or negatively (Brown et al., 1999), but the developer should bear in mind the factors listed in item 1.2.
1.4	**Software control**	Computer coded control of characteristics, e.g. velocities interpreted from physical input devices. These factors need to be programmed for each viewpoint set up. See also item 2.2.3 – collision boundaries
1.4.1	Degrees of freedom	Should represent the degrees of freedom afforded by the equivalent experience in the real world. But see proviso in item 1.4.5.
1.4.2	Linear velocity	Should represent the linear velocity afforded by the equivalent experience in the real world. But see proviso in item 1.4.5.
1.4.3	Angular velocity	Should represent the angular velocity afforded by the equivalent experience in the real world. But see proviso in item 1.4.5.

1.4.4	Axes of rotation	Should afford rotation about the same axes as the equivalent experience in the real world.
1.4.5	Expected user population requirements	Reduce DOF, or linear or angular velocities to compensate if expected user population is likely to have problems navigating due to, for example, over sensitivity of the software control characteristics set up in items 1.4.1 to 1.4.4.
2	**ENVIRONMENT CRITERIA**	
2.1	**Recognition**	Different types of VE require different levels of object/location recognition. However, there is a minimum requirement that a user can recognise, for example, an object as an obstacle, even if they do not know exactly what type of object that obstacle is supposed to be. See also item 1.2 – choice of display device
2.1.1	Fidelity	Fidelity allows recognition of objects, which may, in turn, give the user information about that object's affordances re navigation (e.g. a door affords movement to a different room)
2.1.2	Landmarks	Landmarks can support recognition of a location and help build a conceptual map of a virtual space (Steck & Mallot, 2000). By giving them a directional element they can also provide information about orientation.
2.2	**Negotiating the topography**	For the user to move about the VE there needs to be negotiable routes that they can use
2.2.1	Passage ways	Widen passage ways (e.g. doorways or corridors) to make navigation easier (Neale et al., 2000).
2.2.2	Obstacles	Remove obstacles (where they are of no benefit to the VE) to make navigation easier
2.2.3	Collision boundaries	Objects should afford the expected resistance to impact by users (or other moving objects).
2.3	**Way finding**	Users' effective use of a VE will usually be improved by them building up an understanding of the VE's layout (Darken & Sibert, 1996).
2.3.1	User aids	Devices (2D or 3D) can be used to improve the user's understanding of the topography of a VE, for example; arrows, maps, grid, compass, emphasised structure (e.g. colour coding of areas, obvious paths, clearly visible goals) (R. P. Darken & W. P. Banker, 1998).
2.3.2	Jump to location	Preset viewpoints accessible for example, via on-screen buttons or keyboard key press. Used where comprehension of the topology is not required, but instant access to locations is desirable.

Table 6.2. The navigation guidance tool

The tool is split into two sections, interface criteria and environment criteria, as per the model described in section 3.6.1 and figure 3.12. The interface criteria covered by the tool are; the choice of hardware input devices, the choice of display device, the layout of the display, and software control. The environment criteria are recognition, negotiating the topography, and way finding. Each of these criteria will now be discussed in more detail.

The choice of hardware input and display devices is likely to be made at the specification stage of VEDS as part of the VR system configuration (figure 6.1). It is therefore likely that when it comes to developing the navigation interface, the VE developer will have to work with whatever devices have been specified. This choice of devices will have an impact on decisions made by the developer later on, especially with respect to the DOF simultaneously afforded by the input device, and the image quality afforded by the display system. However, whilst some VEs require the manoeuvrability of six DOF and/or a high quality image display (e.g. 24 bit colour, greater than 1024x768 resolution). at the other extreme there are VEs where the navigation modes only require two DOF and, because of the simplicity of the models, the image renders adequately using a colour depth of 256 colours on a 640x480 (or lower) resolution display. In the later case, increases in the display resolution and colour depth used are likely to affect the image refresh rate adversely, regardless of whether the visual appearance of the VE actually benefits.

The use of hybrid displays (the combination of a window onto a 3D environment, and 2D elements such as buttons and text boxes) brings with it the problem of how best to arrange the screen layout. Often the 2D elements are navigation aids (e.g. maps, jump to location buttons or the Superscape 'move bar') and their position on the screen will impact on usability. There has been very little research done into these types of interfaces (see Brown (1999)), but it would seem logical to suggest that grouping navigational aids separately from other 2D elements will aid user comprehension. Hybrid displays will usually incorporate a window onto the 3D environment that is smaller than the full screen size, to make room for the 2D elements. Increasing or reducing this window size will affect the image resolution (in terms of the total number of pixels used) with the corresponding effect on the level of detail that can be perceived (see item 1.2.1 in table 6.2).

The final interface criterion that needs to be considered by the VE developer is the software control of the navigation characteristics. Here the developer should be aiming to match the capabilities of the input device(s) to the real world navigational characteristics of the experience being modelled. This has to be done with the understanding that the user may not have the skills required to perform these activities in the real world, let alone via the limitations of a VR user interface. The development process therefore, is one of initially attempting to make the experience similar to the equivalent experience navigating in the real world. Then, if that configuration is judged too difficult for the expected user population to control, it can be made easier by reducing the linear and/or angular velocities interpreted from the input device, and/or the number of simultaneous DOF available.

In order to successfully navigate within a VE, the user needs to be able to recognise features within the VE, develop a plan of where they want to go, and negotiate a path to get there (Ruddle et al., 1998). From the VE developer's point of view, these requirements are likely to be considered in a different order as dictated by the VE development process, the objects being modelled first (recognition), followed by the layout of the VE (negotiation), and finally way-finding tools may be added as required.

The level of object recognition required in a VE depends on the role of that object within the VE. From the user's point of view, whilst navigating, they will need to know whether an object is for example, a boundary, an obstacle or a pathway. This recognition requires a fairly low LOD although other fidelity requirements on the same objects may result in them having a much higher LOD.

Landmarks can be used to aid recognition of specific locations within a VE. Often a landmark is an object that already exists in a VE at the required location but, if necessary, it can be made more distinct so that when it is recognised it will help the user to remember where they are (assuming they have been there before). Alternatively, landmarks can be added to a VE where no suitable objects already exist. Landmarks can be further enhanced by making them recognisable from any angle from which the user is likely to view them. They can also be given a directional element so that they provide information about orientation as well as location.

For the user to move about the VE, negotiable routes are required. Some VEs naturally lend themselves to easy negotiation but in other cases the topography may need to be adapted to allow for the reduced agility provided by VR system interfaces. Apart from removing obstacles, widening passageways is another way in which movement through a VE can be made easier (Neale et al., 2000).

Giving objects correct collision boundaries can hinder movement through a VE as it forces the user to use the conventional routes through the environment as they would in the real world. On the other hand, removing the collision boundaries will result in an unrealistic experience, and can lead to the user becoming lost in cyberspace due to the confusion caused by moving between locations without being forced to follow a 'path'. The author would therefore recommend that, as the norm, collision boundaries are used unless there is good reason not to do so. An example of when not to use collision boundaries is; if an object forms an obstacle to navigation, but is necessary to fill another VE requirement such as fidelity. However, it should be taken into consideration that the removal of the collision boundary is itself a reduction of fidelity. Sometimes it is possible to reduce the size of an object's collision boundary such that it facilitates navigation, whilst the user walking straight at the object still results in a collision. This maintains most of the fidelity whilst improving navigation.

Planning a route through an unexplored environment will usually benefit from some form of way-finding aid. In a VE there are many forms these aids can take and many ways of presenting them. Sometimes they can be implemented in a naturalistic way as suggested by Fencott (In preparation), whereby a path or a handrail indicate a negotiable route. They could be superimposed upon the 3D environment in the form of arrows or grid lines. Alternatively they could be added as part of a hybrid display system in the form of a 2D map with a dynamic 'you are here' indication. Where comprehension of a VE's topology is not important, instant access to locations can be provided via for example, on screen buttons (on a hybrid display, see section 5.2.2, figures 5.1 and 5.2). Alternatively a 3D metaphor such as a lift could be used to 'transport' the user to the new location (see section 5.3.4 and figure 5.26). In each case the destination should be indicated at the point of instigating the 'jump to location', either graphically or textually.

6.5.3 Object number reduction & level of detail control

Chapter three (section 3.6.3) includes an extensive discussion on object number reduction techniques. This section attempts to consider these techniques, together with assessments of the LOD required, to form a concise guidance tool that could be used during the building stage of VE development. Object number reduction is one of the most important issues in VE development. It is also possibly one of the most overlooked by VE developers and researchers alike. By carefully deciding which objects to omit, the VE developer can significantly reduce VE development time without significantly reducing the effectiveness of the VE. There are other benefits such as the increased rendering speed (less facets to render) and even the reduced file size. The user can also benefit from having a simpler, less cluttered, less confusing VE. Omitting the wrong objects, however, could have disastrous consequences for the effectiveness of the VE. Table 6.3 is a checklist that the developer can use when deciding whether to model an object. The left hand column is a list of qualities of object that should be included in a VE. The right hand column suggests the minimum LOD that should be used for an object of that type. As before, if an object has more than one function in a VE, the LOD applied should be the highest one of those suggested for each of its functions. It is important to remember that the minimum level of detail to be applied has to be considered with reference to any anticipated display device such that the criteria will be met when the VE is in use on the target system. Many objects could end up being modelled with excessive LOD. This could happen because an object is imported from another VE where it performed a different function, or because the developer built in a 'safety margin' of detail in case an object's function changed or in case other factors reduce the effective fidelity of the VR system. Provided this does not excessively increase development time, and that a sufficiently high rendering speed is maintained, this is not a problem. Ultimately the LOD applied to an object will be left, initially at least, to the discretion of the VE developer as opinions on what, for example, is 'recognisable' can be very subjective.

Model an object if it is one or more of the following …	**LOD applied should be sufficient to make the object...**
A landmark – any distinct object used to mark a location	Distinct
A boundary – a barrier between two spaces, e.g. a wall	Visible
A device - any object that affords interaction	Recognisable
A feature – an object specific to the environment that helps to inform the user about the nature of the environment. Not all features need to be modelled, in many situations one or two give sufficient information. E.g. a filing cabinet in an office	Recognisable
A cue – something that indicates affordances, e.g. an on/off switch	Comprehensible
A scale informant – an object that has a (approximate) set size regardless of its context (e.g. a chair), and that can therefore be used (subconsciously) by a viewer to gauge the scale of neighbouring objects. Not all scale informants need to be modelled, in many situations one or two give sufficient information	Recognisable and accurate
A pathway – a continuous visual reference showing the route from one place to another, e.g. a road or a handrail	Recognisable
An aid – anything put into the VE to provide information not immediately implicit from the geometry, e.g. a list of instructions, map, signpost etc	Comprehensible

Table 6.3. Object number reduction checklist tool

In order to perform its function as a landmark, an object needs to be distinct, i.e. visually different from all other objects in the VE. A boundary needs to be visible such that the user understands why movement in a specific direction is not possible. A device needs to be recognisable so that the user can tell that it is a device. If comprehension is also required (i.e. the device incorporates one or more cues) then an increased LOD will be necessary. In order to successfully help to indicate the nature of an environment, a feature will need to be recognisable. A cue needs to be comprehensible, i.e. the user needs to be able to use a cue or cues to tell what an object is, what its function is, and how to use it. A scale informant has firstly to be recognisable, so that the user can tell what type of object it is, and secondly it has to be modelled with sufficient accuracy of size to give the user the required scale information. A pathway is used by the VE designer to encourage the user to travel along a specific route. There is some evidence that users will follow any (horizontal) line and use it as a means of navigating (Darken & Sibert, 1996) so it may not be essential to make pathways recognisable. It would seem sensible to suggest however, that a user would be more likely to use a pathway that looked, for example, like a road. An aid is a bit like a cue in many respects, in that it is used give the

user information about the VE. Some cues will be aids and vice versa. Whereas cues exist within the 3D environment and tend to be naturalistic, aids are often unrealistic additions to the 3D environment, or appear outside the environment as 2D elements in a hybrid display. Aids can also be much more extensive than a cue, giving much more complex information about the topography or function of part of, or the whole of, a VE. This requirement to provide complex information leads to the need for aids to be comprehensible to the user.

6.6 Conclusions

This chapter has shown how VEDS was reworked so that it would more accurately represent the specification, design, building and implementation phases of VE development. This structure was revised from the experience gained whilst developing desktop VEs. Most of the processes described here are common to VE development for the full range of types of VR platforms (e.g. stereo HMD or CAVE systems). It should be noted, however, that some platforms will have their own additional specific development requirements not covered here, for example, with respect to implementing stereoscopy. This potential for generalisation will also be largely true for the three guidance tools. Object number reduction and LOD control are processes that will be necessary in all VE development, regardless of the platform. With respect to interaction and navigation, the use of 2D, non-naturalistic elements is less effective in immersive systems, but the need for such enhancements (for interaction at least) is often reduced by the increased flexibility of the hardware user interface. For example, a system with two cyber gloves and an HMD (all tracked) will facilitate interactions not possible using a desktop system controlled by a joystick and a mouse.

Chapter 7 Conclusions and Recommendations for Further Research

There could be many different ways of viewing VR/VE technology. At one extreme there is the coding and algorithms required to render 3D shapes on a computer display. At the other extreme is the users' opinion of a VE, possibly in terms of how useful it is. but ultimately in terms of how much fun it is. This thesis has attempted to look at VR/VE technology in a way that is most useful to VE developers. A VE developer does need to have some understanding of 3D rendering algorithms and how to make a VE 'fun'. as well as having a grasp of many of the spectrum of topics in between these extremes, but more than that, they need to be able to combine and use this comprehension to make sense of the entire VE development process. It was with a view to fulfilling these requirements that this research was undertaken.

This thesis has met its overall aim of "developing structure and guidance for VE development by looking at the application and user requirements in conjunction with the technical options and constraints". Taking each of the specific objectives in turn;

1. A case study approach was used to establish the issues that are most relevant to the VE developer. The issues first confronted were those concerned with VE building and the programming of functionality. In later case studies, more VE usability issues were recognised.

2. The experience gained from the case studies was used to develop models of; the interfaces in VE system use, VE topography, the procedure for creating a virtual object, objects behaviours and dynamics, types of viewpoint and their freedom of movement, the user's VE experience, the hierarchical structure of the components of navigation, and the relationship between cue, feedback and potential for interaction.

3. The existing VEDS was reviewed and found to require significant reworking in order for it to adequately represent the specification, building and implementation phases of VE development. Many of the key processes did not appear at all in the existing structure, whilst others were out of sequence.

4. The results of the review and the previously developed models were used along with the accumulated experiences of VE development to formulate seven new sections to replace the three less detailed ones from the existing VEDS.

5. The 'detail design' phase of VEDS was identified as an area where development guidance would be helpful. This phase is split into seven areas, each of which could benefit from having a design guidance tool. These areas are topography, object shape and appearance, dynamics and behaviours, interactions (methods, cues and feedback). navigation and viewpoints, text boxes and buttons, and object number reduction and level of detail control.

6. Exemplar design guidance tools were developed for three of the seven areas. These were interactions, navigation and viewpoints, and object number reduction and level of detail control.

There are two main ways in which this research could be extended. These are the evaluation of VEDS and the three guidance tools presented in this thesis, and the development and evaluation of further design guidance tools. The order in which this research should be carried out is open to question. However, evaluating the three tools first would allow any lessons learned to be applied in the development of the new tools.

The reworked version of VEDS needs to be evaluated to find out whether this structure is consistent with the successful development methods used by VE developers across a range of VR hardware and VE application types. The three guidance tools need to be evaluated to find out if and how they benefit the VE development process. This benefit could be in the form of, reduced development time or cost, or improved VE utility or usability. They may have other benefits such as making VE developers feel more in control of the VE design process, or helping them explain the process to a client. Additionally, the application of these tools may aid the creation of VEs that are more enjoyable from the users' point of view. To find out if the tools require any adaptation this evaluation also needs to take place across a range of VR hardware and VE application types.

This thesis suggests a possible four further design guidance tools that could benefit the VE development process. These are the proposed tools to provide guidance on designing (1) VE topography, (2) object shape and appearance, (3) object behaviour and dynamics, and (4) hybrid interfaces (text boxes and buttons). Some of the background research for each these tools can be found in this thesis.

VE development is such a complex process that a comprehensive set of guidelines could well be too large to be of any practical use. It would be like trying to explain to somebody (who'd never heard of the game) how to play football. No explanation would on its own, result in them being a good player. However, such an explanation could run into many thousands of pages that would be too long winded to be useful. What footballers do need is a good understanding of the game and lots of practice/experience (and fitness etc). Once they have this, they can start to adopt beneficial strategies and tactics. They also need to be able to communicate well with the rest of their team. The situation is similar with VE design. The VE developer needs to have a good understanding of the issues involved and the ability to communicate those issues to the other people working on the VE development project. They can then apply relevant techniques to the development process. This thesis has attempted to categorise and structure the VE development process in ways that make it more digestible, in the hope that this will enable VE developers to grasp the relevant concepts more quickly. It has also explained a range of VE development techniques and provided guidance tools to advise as to their most appropriate use.

References

Akselsson, K., Bengtsson, P., Eriksson, J., Johansson, C., Johansson, G., & Klercker, J. (1994). Computer Aided Planning of Production and Working Environment. *Human Factors in Organizational IP Design and Management*(IV), 499-504.

Barfield, W., & Furness III, T. (1995). *Virtual Environments and Advanced Interface Design*. Oxford: Oxford University Press.

Bowman, D. A., Johnson, D. B., & Hodges, L. F. (2001). Testbed Evaluation of Virtual Environment Interaction Techniques. *Presence, 10*(1), 75-95.

Bowman, D. A., Kruijff, E., LaViola, J. J. j., & Poupyrev, I. (2001). An Introduction to 3D User Interface Design. *Presence, 10*(1), 96-108.

Boyd, D., & Sastry, L. (1999). Development of the INQUISITIVE Interaction Toolkit - Concept and Realisation. In proceedings of *User Centred Design and the Implementation of Virtual Environments*, York, 1-6.

Brown, D., Cobb, S., & Eastgate, R. (1995). Learning in Virtual Environments (LIVE). In R. Earnshaw & J. Vince & H. Jones (Eds.), *Virtual Reality and its Application* (pp. 245-252). London: Academic Press.

Brown, D., Cope, N., Cobb, S., & Eastgate, R. (1993). *VR as a tool to train operators in compartment layout and to maintain plant efficiently in hazardous environments.*: VIRART, University of Nottingham.

Brown, D., Eastgate, R., & Collins, B. (1994). *Virtual Reality Applications for Factory Planning and Layout* (Internal report for Unilever): VIRART, University of Nottingham.

Brown, D., Neale, H., Cobb, S., & Reynolds, H. (1999). Development and Evaluation of the Virtual City. *The International Journal of Virtual Reality, 4*(1), 28-41.

Bullinger, H., Breining, R., & Bauer, W. (1999). Virtual Prototyping – State of the Art in Product Design. In proceedings of *The 26th International Conference on Computers & Industrial Engineering*, Melbourne, 103-107.

Church, D. (1999). Formal Abstract Design Tools. *Games Developer Magazine, August*.

Cobb, S. (1999). Measurement of postural stability before and after exposure in a virtual environment. *Applied Ergonomics, 30*, 47 - 57.

Cobb, S., Brown, D., Eastgate, R., & Wilson, J. (1993). Learning In Virtual Environments (LIVE). In proceedings of *Science for Life*, University of Keele.

Cobb, S., Neale, H., Crosier, J., & Wilson, J. (2000). Development and Evaluation of Virtual Environments for Education. In K. Stanney (Ed.), *Handbook of virtual environments*.

Cobb, S., Neale, H., & Reynolds, H. (1998). Evaluation of Virtual Learning Environments. In proceedings of *The 2nd European Conference on Disability, Virtual Reality and Associated Technologies (ICDVRAT)*, Skovde, Sweden. University of Reading, 17-23.

Cobb, S., & Nichols, S. (1996). *Measurement of postural stability before and after immersion in a VE* (VIRART Report 96/130). Nottingham: University of Nottingham.

Cobb, S., & Nichols, S. (1998). Static posture tests for the assessment of postural instability after virtual environment use. *Brain Research Bulletin, 47*(5), 459-464.

Cobb, S., Nichols, S., Ramsey, A., & Wilson, J. R. (1999). Virtual Reality-Induced Symptoms and Effects (VRISE). *Presence, 8*(2), 169-186.

Cobb, S., Nichols, S. and Wilson, JR. (1995). (1995). Health and Safety Implications of Virtual Reality: In Search of an Experimental Methodology. In proceedings of *Proceedings of the FIVE '95. Framework for Immersive Virtual Environments.*, QMW University of London, UK., 227 - 242.

Crosier, J. (1996). *Experimental comparison of different input devices into virtual reality systems for use by children with severe learning difficulties.* Unpublished undergraduate thesis, University of Nottingham, Nottingham.

Crosier, J. (2000). *Virtual environments in science education: a schools-based development.* Unpublished PhD thesis, University of Nottingham, Nottingham.

Darken, R., & Sibert, J. (1996). Navigating Large Virtual Spaces. *International Journal of Human-Computer Interaction, 8*(1), 49-72.

Darken, R. P., & Banker, W. P. (1998). Navigating in Natural Environments: A Virtual Environment Training Transfer Study. *IEEE Annual Virtual Reality International Symposium 1998*, 12-19.

Davies, R. (2000). *Using Virtual Reality for Participatory Design and Brain Injury Rehabilitation.* Unpublished PhD, Lund Institute of Technology, Lund, Sweden.

Davis, S. B., & Athoussaki, H. (1997). VRML:A Designer's View. *Virtual Worlds on the Internet* (pp. 35-51).

D'Cruz, M. (1999). *Structured evaluation of training in virtual environments.* Unpublished Doctor of Philosophy, Nottingham, Nottingham.

D'Cruz, M., Eastgate, R., & Wilson, J. (1997). A study into the issues involved when applying virtual environment technology to training applications. In proceedings of *Virtual Reality WorldWide '97*, Santa Clara, California.

Dickens, P., Cobb, R., Gibson, I., & Pridham, M. (1993). Rapid Prototyping using 3D Welding. *Journal of Design and Manufacturing, 3*(1), 39-44.

Eastgate, R., D'Cruz, M., & Wilson, J. (1995). *Desktop Virtual Reality as a tool to enhance the replenishment and maintenance process of an Automated Teller Machine (ATM)* (Report for AT&T Global Information Solutions): VIRART, University of Nottingham.

Eastgate, R., D'Cruz, M., & Wilson, J. (1997). A Strategy for Interactivity within Virtual Environment Applications : A Virtual ATM Case Study. In proceedings of *Virtual Reality WorldWide '97*, Santa Clara, California.

Eastgate, R., Nichols, S., & D'Cruz, M. (1997). Application of human performance theory to virtual environment development. In D. Harris (Ed.), *Engineering Psychology & Cognitive Ergonomics (Volume 2 - Job Design and Product Design),* (pp. 467 - 475.). Ashgate: Aldershot.

Eastgate, R., & Wilson, J. (1994, December). Virtual Worlds. *EXE: The Software Developers' Magazine, 9,* 14 - 16.

Ebbesmeyer, P., Gehrmann, P., Grafe, M., & Krumm, H. (1999). Virtual Reality for Power Plant Design. *ASME DTEC Computers in Engineering 99.*

Ellis, S. (1995). Origins and elements of virtual environments. In W. Barfield & T. Furness III (Eds.), *Virtual Environments and Advanced Interface Design* (pp. 14-57). Oxford: Oxford University Press.

Endsley, M. (1995). Toward a theory of situation awareness in dynamic systems. *Human Factors, 37*, 32 - 64.

Eyles, D. (1991). A computer graphics system for visualising spacecraft in orbit. In S. Ellis (Ed.), *Pictorial communication in virtual and real environments* (pp. 196-206). London: Taylor & Francis.

Fencott, C. (In preparation). A Practical Content Model for Virtual Environment Design. *International Journal of Human-Computer Studies.*

Fernando, T., Fa, M., Dew, P., & Munlin, M. (1995). Constraint-based 3D manipulation techniques within virtual environments. In R. Earnshaw & J. Vince & H. Jones (Eds.), *Virtual Reality Applications* (pp. 71-89). London: Academic Press.

Fernando, T., Murray, N., Tan, K., & Wimalaratne, P. (1999). Software Architecture for a Constraint-based Virtual Environment. In proceedings of *ACM International Symposium on Virtual Reality Software and Technology, VRST 99*, London. UK.

Gabbard, J., & Hix, D. (1997). *A taxonomy of usability characteristics in virtual environments* (Technical Report http://csgrad.cs.vt.edu/~jgabbard/ve/taxonomy): Virginia Polytechnic Institute and State University.

Gabbard, J., Hix, D., & Swan, J. (1999). User-centred design and evaluation of virtual environments. *IEEE Computer Graphics and Applications, 19*, 51 - 59.

Gibson, I., Brown, D., Cobb, S., & Eastgate, R. (1993). Virtual reality and rapid prototyping. In K. Warwick & J. Gray & D. Roberts (Eds.), *Virtual Reality in Engineering* (pp. 51-63). London: IEE.

Gibson, W. (1984). *Neuromancer*. New York: ACE books.

Goldstein, I. (1993). *Training in organisations: needs assessment, development and evaluation* (Third ed.). Pacific Grove, CA: Brooks/Cole Publishing Company.

Greenhalgh, C. (1997). *Large Scale Collaborative Virtual Environments*. Unpublished PhD, University of Nottingham, Nottingham.

Greenhalgh, C., Purbrick, J., Benford, S., Craven, M., Drozd, A., & Taylor, I. (2000). Temporal links: recording and replaying virtual environments. In proceedings of *Multimedia 2000*, Los Angeles, California, USA.

Haines, H. (In preparation). *A Critical Appraisal of Participatory Ergonomics*. Unpublished PhD, University of Nottingham, Nottingham.

Haines, H., & Wilson, J. (1998). *Development of a framework for participatory ergonomics*. ISBN 0 7176 1573 1: HSE Books.

Herndon, K. P., Van Dam, A., & Gleicher, M. (1994). Workshop on the Challenges of 3D Interaction. *SIGCHI Bulletin, 26*(4).

Hutchins, E. (1995). *Cognition in the Wild*. Cambridge, MA: MIT Press.

Istance, H., & Hand, C. (1998). Individual Differences in Navigating Virtual Environments: Navigation Aids in Perspective. In proceedings of *The First International Conference on Usability Evaluation for Virtual Environments*, De Montfort University, Leicester, UK, 65-68.

Kaur Deol, K., Steed, A., Hand, C., Istance, H., & Tromp, J. (2000). Usability evaluation for virtual environments: Workshop at De Montfort University, Leicester, UK. *Interfaces*(44), 4-7.

Kaur, K. (1998). *Designing virtual environments for usability.* Unpublished Doctor of Philosophy, City University, London.

Kessler, G. D., Bowman, D. A., & Hodges, L. F. (2000). The Simple Virtual Environment Library: An Extensible Framework for Building VE Applications. *Presence, 9*(2). 187-208.

Krueger, M. (1983). *Artificial Reality.* Wokingham, U.K.: Addison-Wesley Publishing Company.

Mason, C. A. (1996). Desktop virtual reality in the RAF. In proceedings of *Virtual Reality - User Issues Colloquium*, Cheltenham, UK, Institute of Electrical Engineers, 8/1-8/3.

Mine, M. R., Brooks, F. P., & Sequin, C. H. (1997). Moving Objects in Space: Exploiting Proprioception In Virtual-Environment Interaction. *Computer Graphics Proceedings, Annual Conference Series*, 19-27.

Morrissey, M. (1996). Virtual Reality - An Industry User's Perspective. In proceedings of *Proceedings of Virtual Reality - User Issues.*, Cheltenham and Gloucester College of Higher Education, IEE, 7/1 - 7/2.

Neale, H., & Brown, D. (1998). Usability Evaluation of the Virtual City. In proceedings of *The First International Workshop on Usability Evaluation for Virtual Environments*, De Montfort University, Leicester, UK, 108-111.

Neale, H., Brown, D., Cobb, S., & Wilson, J. (1999). Structured evaluation of virtual environments for special needs education. *Presence: Teleoperators and Virtual Environments, 8*(3), 264 - 282.

Neale, H., Cobb, S., & Wilson, J. (2000). Designing virtual learning environments for people with learning disabilities: usability issues. In proceedings of *ICDVRAT 2000*, Alghero, Sardinia, University of Reading, 265-272.

Newman, W., & Lamming, M. (1995). *Interactive Systems Design.* Reading, Mass.: Addison-Wesley.

Nichols, S. (1999a). Physical ergonomics of virtual environment use. *Applied Ergonomics*(30), 79-90.

Nichols, S. (1999b). *Virtual Reality Induced Symptoms and Effects (VRISE): Methodological and Theoretical Issues.* Unpublished PhD thesis, University of Nottingham, Nottingham.

Nichols, S., Cobb, S., & Wilson, J. (1997). Health and Safety implications of virtual environments: measurement issues. *Presence: Teleoperators and Virtual Environments, 6*(6), 667 - 675.

Nichols, S., Haldane, C., & Wilson, J. (1998). Measurement of presence and its consequences in virtual environments. *International Journal of Human-Computer Studies, 52*(3), 471 - 491.

Nichols, S., Ramsey, A., Cobb, S., Neale, H., D' Cruz, M., & Wilson, J. (2000). *Incidence of Virtual Reality Induced Symptoms and Effects (VRISE) in desktop and projection screen display systems* (Contract Research Report 274/2000): Health and Safety Executive.

Nielson, J. (1993). *Usability Engineering*. San Diego: Academic Press.

Norman, D. (1988). *The Psychology of Everyday Things*. New York: Basic Books.

Poupyrev, I., Weghorst, S., Billinghurst, M., & Ichikawa, T. (1998). Egocentric Object Manipulation in Virtual Environments: Empirical Evaluation of Interaction Techniques. *Computer Graphics Forum, EUROGRAPHICS'98 issue, 17*(3), 41-52.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Hooland, S., & Carey, T. (1994). *Human-Computer Interaction.*: Addison-Wesley Longman Ltd.

Ramsey, A. (1999). *Virtual Reality Induced Symptoms and Effects (VRISE): A psycho-physiological perspective.* Unpublished PhD thesis, University of Nottingham, Nottingham.

Rasmussen, J. (1986). *Information Processing and Human-Machine Interaction.* Amsterdam: North-Holland.

Reason, J. (1990). *Human Error*. Cambridge: Cambridge University Press.

Reddy, M. (1997). *Perceptually Modulated Level of Detail for Virtual Environments.* Unpublished PhD, University of Edinburgh, Edinburgh.

Ruddle, R., Payne, S., & Jones, D. (1998). Navigating Large-Scale "Desk-Top" Virtual Buildings. *Presence, 7*(2), 179-192.

Sanders, M., & McCormick, E. (1993). *Human Factors in engineering and design* (7th edition ed.). New York: McGraw-Hill.

Schaaf Jr., W. (1998). The Boundaries Between Virtual Environment Evaluation and the Research Methods of Other Disciplines. In proceedings of *The First International Workshop on Usability Evaluation for Virtual Environments*. De Montfort University, Leicester, UK, 129-132.

Sholl, M. (1993). The effect of visual field restriction on spatial knowledge acquisition. In proceedings of *Thirty-fourth Annual Meeting of the Psychonomic Society*. Washington, D.C.

Smith, S., Duke, D., & Willans, S. (2000). Designing world objects for usable virtual environments. In proceedings of *Workshop on Design, Specification and Verification of Interactive Systems: DSV-IS 2000*, Limerick, Ireland, 309-319.

Snowden, D. N., West, A. J., & Howard, T. L. J. (1993). Towards the next generation of Human-Computer Interface. In proceedings of *Informatique'93: Interface to Real & Virtual Worlds*, Montpellier, France, 399-408.

Stanney, K. M., Mourant, R. R., & Kennedy, R. S. (1998). Human Factors Issues in Virtual Environments. *Presence, 7*(4), 327-351.

Steck, S., & Mallot, H. (2000). The Role of Global and Local Landmarks in Virtual Environment Navigation. *Presence, 9*(1), 69-83.

Sutcliffe, A. (1995). *Human-Computer Interface Design* (2nd edition ed.). London: McMillan.

Sutherland, I. (1965). The Ultimate Display. In proceedings of *IFIPS Congress*, New York, 506 - 508.

Tromp, J., & Fraser, M. (1998). Designing flow of interaction for virtual environments. In proceedings of *Proceedings of the First International Workshop on Usability Evaluation for Virtual Environments*, Leicester, UK, 162 - 170.

Waller, D. (1999). Factors Affecting the Perception of Interobject Distances in Virtual Environments. *Presence, 8*(6), 657-670.

Waller, D., Hunt, E., & Knapp, D. (1998). The transfer of spatial knowledge in virtual environment training. *Presence: Teleoperators and Virtual Environments,, 7*(2), 129 - 143.

Wann, J., & Mon-Williams, M. (1996). What does virtual reality really NEED?: human factors issues in the design of three-dimensional computer environments. *International Journal of Human-Computer Studies, 44*, 829 - 847.

Wickens, C., & Baker, P. (1995). Cognitive issues in virtual reality. In W. Barfield & T. F. III (Eds.), *Virtual environments and advanced interface design.* (pp. 514 - 541). New York: Oxford Press.

Wickens, C., Gordan, S., & Liu, Y. (1998). *An introduction to Human Factors Engineering*. New York: Longman.

Wilson, J. (1996). Effects of participating in virtual environments: a review of current knowledge. *Safety Science, 23*(1), 39 - 51.

Wilson, J. (1997). Virtual environments and ergonomics: needs and opportunities. *Ergonomics, 40*, 1057 - 1077.

Wilson, J., Brown, D., Cobb, S., D'Cruz, M., & Eastgate, R. (1995). Manufacturing Operations in Virtual Environments (MOVE). *Presence, 4*(3), 306-317.

Wilson, J., Cobb, S., D'Cruz, M., & Eastgate, R. (1996). *Virtual Reality for Industrial Application: Opportunities and Limitations.* Nottingham: Nottingham Academic Press.

Wilson, J. R., Eastgate, R., & D'Cruz, M. (2001). Structured development of virtual environments. In K. Stanney (Ed.), *Handbook of virtual environments.*

Wirth, H., Sokolewicz, M., Bohm, K., & John, W. (1995). MuSE: using VR in system development and validation. In R. Earnshaw & J. Vince & H. Jones (Eds.), *Virtual Reality Applications* (pp. 209-229). London: Academic Press.

Zeltzer, D. (1992). Autonomy, Interaction and Presence. *Presence: Teleoperators and Virtual Environments,, 1*(1), 127 - 132.