VRBridge:
A Constructivist Approach to Supporting Interaction Design and End-User Authoring in Virtual Reality

By

Cara Winterbottom

Thesis Presented for the Degree of

DOCTOR OF PHILOSOPHY

In the Department of Computer Science

UNIVERSITY OF CAPE TOWN

February 2010

Supervised by

Edwin Blake

James Gain

Abstract

For any technology to become widely-used and accepted, it must support end-user authoring and customisation. This means making the technology accessible by enabling understanding of its design issues and reducing its technical barriers. Our interest is in enabling end-users to author dynamic virtual environments (VEs), specifically their interactions: player interactions with objects and the environment; and object interactions with each other and the environment. This thesis describes a method to create tools and design aids which enable end-users to design and implement interactions in a VE and assist them in building the requisite domain knowledge, while reducing the costs of learning a new set of skills.

Our design method is based in constructivism, which is a theory that examines the acquisition and use of knowledge. It provides principles for managing complexity in knowledge acquisition: multiplicity of representations and perspectives; simplicity of basic components; encouragement of exploration; support for deep reflection; and providing users with control of their process as much as possible. We derived two main design aids from these principles: multiple, interactive and synchronised domain-specific representations of the design; and multiple forms of non-invasive and user-adaptable scaffolding. The method began with extensive research into representations and scaffolding, followed by investigation of the design strategies of experts, the needs of novices and how best to support them with software, and the requirements of the VR domain. We also conducted a classroom observation of the practices of non-programmers in VR design, to discover their specific problems with effectively conceptualising and communicating interactions in VR.

Based on our findings in this research and our constructivist guidelines, we developed VRBridge, an interaction authoring tool. This contained a simple event-action interface for creating interactions using trigger-condition-action triads or Triggersets. We conducted two experimental evaluations during the design of VRBridge, to test the effectiveness of our design aids and the basic tool. The first tested the effectiveness of the Triggersets and additional representations: a Floorplan, a Sequence Diagram and Timelines. We used observation, interviews and task success to evaluate how effectively end-users could analyse and debug interactions created with VRBridge. We found that the Triggersets were effective and usable by novices to analyse an interaction design, and that the representations significantly improved end-user work and experience. The second experiment was large-scale (124 participants) and conducted over two weeks. Participants worked on authoring tasks which embodied typical interactions and complexities in the domain. We used a task exploration metric, questionnaires and computer logging to evaluate aspects of task performance: how effectively end-users could create interactions with VRBridge; how effectively they worked in the domain of VR authoring; how much enjoyment or satisfaction they experienced during the

process; and how well they learned over time. This experiment tested the entire system and the effects of the scaffolding and representations.

We found that all users were able to complete authoring tasks using VRBridge after very little experience with the system and domain; all users improved and felt more satisfaction over time; users with representations or scaffolding as a design aid completed the task more expertly, explored more effectively, felt more satisfaction and learned better than those without design aids; users with representations explored more effectively and felt more satisfaction than those with scaffolding; and users with both design aids learned better but did not improve in any other way over users with a single design aid. We also gained evidence about how the scaffolding, representations and basic tool were used during the evaluation.

The contributions of this thesis are: an effective and efficient theory-based design method; a case study in the use of constructivism to structure a design process and deliver effective tools; a proof-of-concept prototype with which novices can create interactions in VR without traditional programming; evidence about the problems that novices face when designing interactions and dealing with unfamiliar programming concepts; empirical evidence about the relative effectiveness of additional representations and scaffolding as support for designing interactions; guidelines for supporting end-user authoring in general; and guidelines for the design of effective interaction authoring systems in general.

Acknowledgements

I would like to thank my supervisors, Edwin Blake and James Gain, for their guidance and support during my PhD. I was funded by the CAVES project and the University of Cape Town and thank them for their financial support.

The Methodology Group and the Multimedia Education Group conducted the study of which my classroom observation was part. I thank them for their insights and work. I would particularly like to thank David Nunez, Hendranus Vermuelen, Charlene Elliott and Gary Marsden.

My colleagues in the Collaborative Visualisation Laboratory of the Department of Computer Science have accompanied me on this journey, some from the very beginning. We have supported each other with serious discussion and light relief, and they have helped to make my journey an enjoyable one. I would like to specifically thank Ilda Ladeira, Sarah Brown and Rudy Neeser.

My parents, Rosemary and Peter Winterbottom, have always given me unqualified support and encouragement. I thank them for their love and belief in me.

Finally, and most importantly, I would like to thank my partner, Andrew Higson-Smith. He has been with me during my entire journey, and has stood by me, supported me and offered advice when needed. I thank him for his love, patience and wisdom.

Glossary

- *Authoring*: The process of designing *and* implementing the design.
- *Authoring Tool:* A tool which assists the user in both creating a design and implementing that design.
- *Design Aid*: A feature that supports design work, by making it easier and/or helping the development of expertise in design.
- *End-user, User and Player*: In this work we make a distinction between the user of VRBridge and the user of the VE that is created. The user of VRBridge is our *end-user*, the designer who wants to create a VE for others. The user of the VE that is created by our end-users is the *player.* In our text, we use the terms *user* and *end-user* to denote the designer working with VRBridge. To denote the user of the VE, we use the term *player* in our text and the term *User* (with an initial capital) in the VRBridge software.
- *Fading*: Term describing the way scaffolding falls away when it is no longer useful.
- *Interaction:* Interactions are the relationships between the player of a VE, its objects and the environment itself. Apart from actors (active objects) in the VE, which can interact with the player in different ways depending on time, place and sequence, the environment can also react to the player and other objects, e.g., with changes of scenery and lighting.
- *Scaffolding*: The provision of support to novices, which allows them to do work which they would not otherwise be able to accomplish, that will fall away or fade as novices learn and become more accomplished.
- *Virtual Reality (VR):* A 3D computer-generated world, or virtual environment (VE), which the player explores from a first-person perspective, and with which the player interacts to achieve effects.

Table of Contents

List of Tables

List of Figures

1. Introduction

The distinction between end-users and authors is lessening in the area of computer applications. In recent years we have seen the advent of technologies such as blogging software, presentation software, game level editors and video and sound editing programs, which allow those with very little technical knowledge or creative training to design, construct and disseminate creative products. The end-user becomes more of a participant in the creative process, rather than just a consumer.

This changing distinction has been discussed by theorists. Myers et al. (2000) highlighted end-user programming and customisation as an important future trend in a review of user interface software tools. The major obstacle that they foresaw was high software thresholds: the difficulty of learning to use such systems. The ideal is a system where the end-user can begin working immediately and learn in small increments, without hitting overwhelming obstacles. Blackwell (2002) highlighted the same problem while considering the nature of programming. He argued for a common understanding of both end-user and professional programming as a cognitive activity that involves loss of direct manipulation, the introduction of specific notations and the consequent increase in abstraction. He then defined the attention investment model for understanding how to foster and support usage of programming systems; a user will weigh up the costs (in effort and resources) of investing attention in a system, against the benefits of being able to create a product or automate a task. Burnett (2009) goes further and considers the quality of end-user products, and how to foster end-user software engineering. She describes the Surprise-Explain-Reward strategy, whereby users' curiosity is aroused so that they are interested in explanations about gaps in their knowledge. These explanations reward their curiosity by providing the benefit of increased knowledge and improved products.

End-users are people who want to use technologies for creative work, but for whom the technological skills are not part of their primary expertise. End-users could have expertise in a range of disciplines, e.g., marketing, education, archaeology and psychology. They will also have a wide variety of programming capability and design experience. This group is large and varied; therefore any system which attempts to support them in creative work must cater to those who are novice designers and programmers, while still being useful for those with more experience.

The implication of this is that computer systems must support creative work by the novice user, and enable fast learning of the interface. Some theorists have considered how to do this generically. Shneiderman (2000), for instance, has considered how computers could support creativity. He suggests supporting creative work via: enabling visualisation of processes and products; providing immediate feedback for exploration of solutions; enabling access to expert knowledge; fostering reflection through supporting effective work evaluation; supporting association of concepts and their relationships; and allowing useful products to be

created. Hewett (2005) reviews the nature of creativity and finds commonalities across domains: creators control their work process and environment; they learn by doing work in the domain and stretching themselves to new insights. Like Shneiderman, he discusses how computers can support creativity through optimising the working environment, leveraging appropriate knowledge and making problem space constraints visible. His proposed methods of support are similar to Shneiderman's: multiple representations of the problem space which can be explored, reflected on and combined; simple capture of solutions; and easy and constant access to context, tool and domain help.

Jonassen and Carr (2000) discuss the computer as a mind-tool, which changes the user's task and provides multiple supports for learning and representing knowledge. They specify various kinds of support provided by computer tools, e.g., dynamic modelling tools which help users to describe dynamic relationships, visualisation tools which help users to reason visually, and semantic organisation tools which help users to organise and analyse their knowledge.

We have been inspired by these theorists to address the above issues in our domain of interest: virtual reality (VR). VR is no longer a novel medium. Its use has been extensively researched in a wide variety of projects, from investigation of application areas, such as phobia treatment (Hoffman et al. 2003) and cultural heritage (Ladeira and Blake 2004), to techniques for interaction with a virtual environment (Bowman et al. 2001) and how to enhance the player's experience (Schuemie et al. 2001). To date, VR has mostly been used as a research tool or for large-scale projects, such as 3D games or artistic installations. Virtual environments (VEs) have the potential to become easily and commonly used; however, in order to accomplish this, VR must be made more accessible. This position is clearly described by the Innovation Diffusion Theory:

"*…technology adoption cannot occur unless the innovation is distributed or diffused through a society. The essence of the technology diffusion process is human interaction in which one person communicates a new idea to another person*." (Gross 2002, p. 534)

From a slightly different perspective, Brenda Laurel, an important contributor to VR theory, states:

"*For virtual reality to succeed in meeting these goals* (intimate and powerful experiences)*, we need continual and deep involvement by artists in the ongoing process of understanding what virtual reality is for and what it can be. We need convivial tools that allow artists to work in the medium in order to influence its evolution*." (Laurel 1993, p.196)

Our research stems from the convergence of the increasing interest in end-user programming and novice creativity support, and the desire to make VR authoring more accessible to end-users. When we began

investigating these areas, we were struck by the similarity between methods for supporting end-user creativity and the principles of the *constructivist theory* of knowledge building (Fosnot 1996). Constructivism focuses on learning through doing. As such, the theory supports notions of multiplicity of representation and explanation, reflection on actions and their consequences, support for exploration, help in the form of scaffolding that can fall away rather than marking out an exact path, and user control of the process. Based on these connections and the wealth of theoretical support provided by constructivism, we decided to use the theory to structure our research. This provides us with a powerful theoretical grounding for understanding how to practically facilitate end-user knowledge building and creativity.

1.1. Research Context

Our interest is in supporting end-user creativity in the authoring of interactive systems, specifically within the domain of VR. We use the word *authoring* to mean both designing and implementing a design. For VR to become accessible, end-users must be able to create products so that they can perform the entire process of creation and take pride in it. There are two tensions at work here: the creative difficulty of designing an interesting and useful product; and the technical difficulty of turning that design into reality. Below, we describe the general domain of designing interactive systems and the domain of virtual reality in more detail, focussing on the creative and technical difficulties that must be addressed during authoring.

1.1.1. Designing Interactive Systems

It is generally recognised that design is difficult. It is made even more intractable by the fact that it can be applied across domains to yield vastly different design artefacts with different requirements. Any account of a design process is necessarily situated within the domain for which the design is required (as will be seen from our account of designing VR interactions below). However, if one examines accounts of expert designers across disciplines, it is possible to find over-arching principles and practices which can be applied to specific design problems. For example, Schön, a seminal theorist who wrote about the process of learning to design architecture, stated:

"A student cannot understand and acquire each component skill, in the sense in which 'thinking architecturally' requires it, until he has experienced that component in the context of a whole process."
(Schön 1987, p. 97)

This provides us with an understanding of the difficulty of design in a domain, as it requires both skill in design and skill in the domain. Learning design means learning both. Schön also indicates two ways in which we can begin to support this process. We must help users to *think* in the style of the domain, i.e., understand its particular requirements and complexities, so that they can begin to understand how to design for the

domain. We must also help users to work actively within the domain as early as possible so that they can *experience* the design process and understand how each part is composed to create a completed design. Designing for interactivity brings additional difficulties, as the designer cannot control the user experience directly. Usage of the designed system emerges from the functioning of its rules. Salen et al. (2003) advocate creating an *interactive prototype* rather than a *visual prototype* in the design of games (which are interactive systems similar to VR) because:

"It is never possible to completely predict the experience of a game…Learning to play a game critically, seeing where it excels and where it grinds to a halt, and being able to implement changes that will push the game toward meaningful play are all core game design skills." (Salen et al. 2003, pp. 11-12).

This advice is pertinent to all interactive design processes. For any interactive system, it is impossible to completely predict the users' actions; working with a prototype that allows the designer to experience the design and think about how it will be used addresses this problem. With any interactive system, from a visualisation tool to a searchable database, increasing the amount, flexibility and impact of interaction increases the power and creative involvement of the end-user (Laurel 1993).

1.1.2. Virtual Reality

The domain of VR is a large one, which encompasses many roles. A complete VE has several components, each of which requires different skills to realise: content must be designed and created e.g., 3D models, textures, animations and sound; the underlying mechanisms for displaying content must be developed e.g., graphics, audio, physics and networking; and the interactions between various objects in the VE and the player must be scripted, using the content and within the constraints of the VR system. We have chosen to focus on the last aspect, interaction authoring, for several reasons.

The first is that successful attempts have been made to address accessibility of the first two aspects. There are many applications available for creating content, which are aimed at a range of users, and much content is freely available on the internet. There are also various engines available, which handle the underlying mechanisms for displaying and interacting with content. For designing interactions, there are fewer options. Those that are available either have limited power (e.g. scripting interfaces for games), require programming ability (e.g. Flash programming), or require financial investment. The second reason is that many theorists in both VR and games have stated that interaction design is the most important aspect of creating a VR, e.g.:

"The focus of a game designer is designing game play, conceiving and designing rules and structures that result in an experience for players." (Salen et al. 2003, p. 1)

"The art of VE design is surely to provide users with carefully structured opportunities to allow them to explore, strategize, and generally feel some sense of control over what they are doing." (Fencott 2001b, p. 27)

"The most brilliant concept is useless unless you can think of the way that people will play the game. In the end, game design comes down to interface design—the key to making games playable is how you'll get people to interact with your concept and how simple the interface is." (Peter Molyneux, designer of Theme Park and Black & White, in Saltzman 2003 p.77)

"Yes, I can do 'anything' in a virtual world, but how does the world respond?" (Laurel 1993, p. 186)

The third reason is that interaction design is difficult, even for experts. Our focus is on helping non-expert users to author effective VEs; therefore we want to support them in designing these VEs as well as removing technical barriers. By highlighting the technical constraints on VR authoring, we make designers aware of what is technically feasible. Therefore, as well as supporting creation of VEs by single users, we can improve communication and understanding between programmers and designers in larger teams.

The interactions of a VE make it dynamic and bring the content to life. Before we examine the particular difficulties with interaction design in VR, we must define more clearly what we mean by interaction in this context. Interactions are the relationships between the player of a VE, its objects and the environment itself. Apart from actors (active objects) in the VE, which can interact with the player in different ways depending on time, place and sequence, the environment can also react to the player and other objects, e.g. with changes of scenery and lighting. It is not difficult to design simple interactions in a VE; the problem lies in designing effective and nuanced interactions which will create a compelling and dynamic experience for the player.

Designing effective interactions in VR is especially difficult because VR presupposes an independent player. Any narrative or goal must be flexible enough to be found, without forcing the player along a path. There are times when it can be effective to control the player (e.g., to create a mood), but the innovation of VR is that the player determines the progression of events. The static environment can guide the player, e.g., perceptual cues can direct attention; however, dynamic events are more effective, and also engage the player directly in the narrative of the VE. There are many additional complexities to consider when designing VR interactions:

- Interactions happen over time for an indeterminate duration, so the design cannot be viewed statically.
- Multiple entities are involved in the action, so that each entity may participate in different sets of interactions. This leads to a combinatorial escalation of possibilities for interaction.
- Interactions include actions that are so commonplace that we do not think about them, like keeping our feet on the floor and avoiding obstacles. They require a high level of detail to define completely.

- The VE must be sufficiently reactive to the player's interactions to make the experience enjoyable and interesting. This includes real-time reactions to player input.
- Many interactions will be dependent on player actions, which cannot be predetermined. Therefore, the designer must deal with a significant amount of uncertainty about how the end result will be experienced.
- The interactions take place in a 3D environment, and require understanding of 3D space. In addition, interactions may be location-based.

Figure 1.1 illustrates some design considerations for the simple action of moving an object towards the player when triggered.

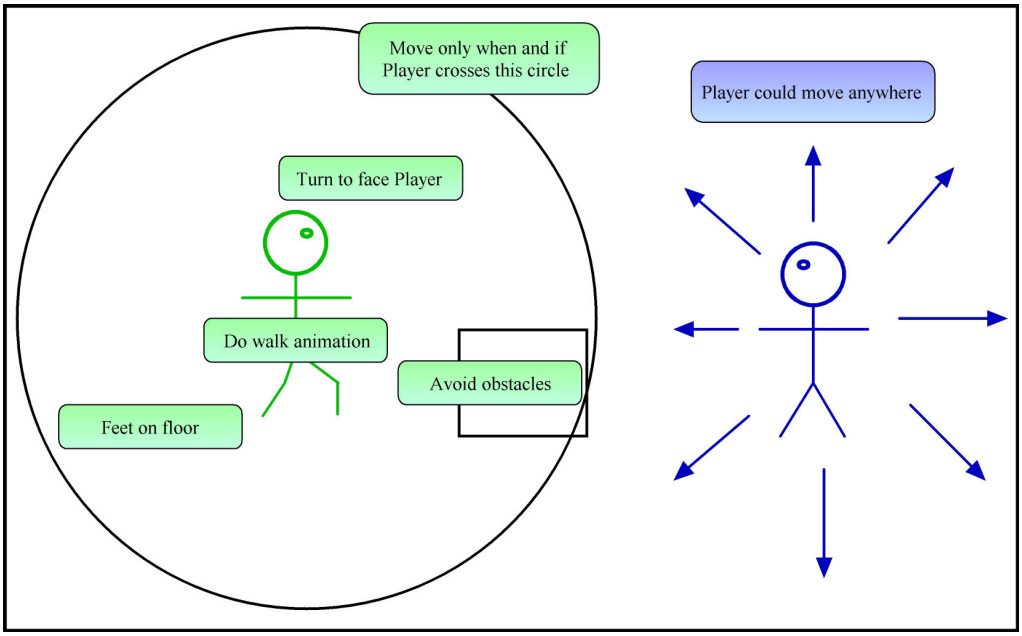


Figure 1.1: Some aspects to be considered for the simple action of an object moving towards the player in a VE.

1.2. Research Aims

For this research, we have two main goals: to create a VR authoring system that is accessible to end-users, which they can use to design and implement a VE; and to assist end-users in creating a highly interactive VE, where the player is more likely to feel part of the action, as this is of primary importance in VR. In achieving these goals, we must not provide too much automation, so that end-users can retain creative control and experience as much of the authoring process as possible. As we have chosen constructivism to structure this research, we want to apply constructivist theory at two levels: to structure our design process in creating an effective tool for end-users; and to structure the end-user design process in the domain of VR authoring. The main aims which guide our research are as follows:

1. To develop and follow a theory-based design method (constructivism) which structures the creation of an interaction authoring tool.
2. To identify and explore key constructivist principles that can practically be applied to an end-user interaction authoring environment.
3. To create a VR interaction authoring tool (VRBridge) for end-users based on the features that we identified in (2).
4. To evaluate VRBridge and compare the features identified in (2) in terms of how effectively they improve tool usage, domain task performance and conceptual understanding.
5. To evaluate how well constructivism serves the design process for a VE authoring tool, and the extent to which we can generalise these findings to other interaction authoring and end-user design tools.

These aims correspond to three formal research questions:

1. *What features would a VE authoring environment have if it were created with constructivist principles?*
2. *What is the relative effectiveness of the discovered features from Question 1 for making a design tool easier to learn to use and fostering conceptual understanding of the VE interaction domain?*
3. *How well does constructivism serve the VE design process, and how far can we generalise to other end-user authoring and interaction design tools?*

VRBridge must enable end-users to design and implement interactions in a VE and assist them in building the requisite domain knowledge, while reducing the costs of learning a new set of skills. The expertise typically required both for designing and implementing a VE and its interactions puts VR out of range of many people. Enabling them to produce creative work within the field will enrich the domain as new ideas can be expressed. Even if an interaction design is only accomplished inexpertly, the provision of a dynamic prototype will still enable better communication between members of a project team. This potential for improved communication within design teams is valuable not just within the field of VR and games, but in all design domains.

1.2.1. Limitations of Research

In this research, our emphasis is on producing a proof-of-concept prototype that provides end-users with the ability to create typical interactions in a VE. Therefore, we do not attempt to address the more high-level complexities of advanced game engines and authoring systems. For example, we do not provide networking functionality and therefore do not consider multi-player VR. We also do not include autonomous agents in our system. Finally, because our aim is to create low-cost options which can be used on the computer systems of any end-user, we restrict our technology to commonly available technology. Specifically, we consider the input devices of keyboard and mouse and the display device of the desktop computer monitor. We do not consider programming for motion tracking or head-mounted displays.

We believe that the above facilities are secondary to a good interaction design and implementation (especially for end-users who are not likely to have access to expensive equipment). In most cases, our system can be extended to incorporate the above functionality, and in the course of our research we briefly discuss some of the above as extensions to current work.

1.3. Overview of Thesis

To accomplish our aims we followed an iterative process, interleaving theoretical work, practical design and implementation, and evaluations. We began by examining constructivism and developing our novel design method. The steps of our method are as follows:

1. Distil practical values from the theory to guide design.
2. Examine processes of expert designers and experts in the domain and connect them to the design values.
3. Conduct research into the practices of novices in the domain area to understand their specific problems.
4. Create a prototype and show it to the target audience. Let them use it. Focus on the design values and their effects in questions afterwards. Then incorporate what has been learned into the artefact. Repeat step 4 until goal is met.

The number of cycles through which step 4 of the design method is iterated depends on user satisfaction, and financial and time constraints. This method has similarities to user-centred design with its emphasis on supporting users (Preece et al. 2002); however it adds the focus on having a unifying theory to structure the design and the distinction between users and experts.

1.3.1. Theoretical Foundations

Following our design method, we extracted the following key constructivist principles on which to focus: user control, atomic simplicity, active exploration, multiplicity and reflection. We used these principles to inform our design at two levels: the creation of a basic authoring tool for end-users; and the creation of design aids to support more expert usage of the tool and work in the domain. For the design aids, we identified two core features that embodied multiple constructivist principles and which we could practically implement:

- multiple interactive and synchronised representations to assist design in the domain; and
- multiple forms of dynamic scaffolding (help which fades as the user learns).

1.3.2. Exploratory Research

Having identified our constructivist features, we conducted research on external representations and scaffolding so that we could understand how best to use them. Both features have an extensive history of

theoretical research and evaluation studies. We also conducted research on the development of expertise and the practices of experts in design, interactive design and VR, so that we could understand the best practices to focus on in supporting novices. As part of this endeavour, we conducted research on how to support novices in general.

Once we had extensively examined the literature, we conducted a classroom observation of novices creating VEs. We observed humanities students engaged in designing 3D games as part of a media course. They were provided with some instruction about useful design aids and VR requirements. Although it was envisaged that the students would use a scripting system to implement their designs, it rapidly became apparent that they would not be able to master the required programming concepts in the time provided. Therefore, their designs were implemented by a programmer. Some additional observations from this study were:

- Participants were willing to experiment with new techniques if their usefulness was apparent.
- The design aids that were most effective were a floorplan and a flowchart; although participants produced inaccurate and linear flowcharts, they understood them well.
- Participants forgot to state all details and behaviour explicitly.
- Participants produced linear designs that were more like walkthroughs than possibilities for alternative interactions.
- Participants did not document their interactions clearly or consistently, so the programmer had to make design choices through lack of information.

1.3.3. Design

Having accomplished our goals of understanding how to support expert behaviour in VR interaction authoring and the problems that novices face, we began designing VRBridge. Before designing multiple representations and scaffolding, we created the basic authoring system that embodied our constructivist principles and allowed end-users to specify interactions without traditional programming. We chose to use the event-action programming paradigm, using natural language and direct manipulation as much as possible, as our research suggested that this would work well for novices. We developed trigger-condition-action triads for authoring interactions, which we called *Triggersets*. Each part of the Triggerset language is simple, but they can be composed into complex interactions. Since they are easy to understand and can be used to create a prototype efficiently, they lessen the need for detailed documentation of a design.

Having created the authoring interface, we designed and implemented an initial version of our representation design aid. We decided on three additional representations: a Floorplan to help understanding of space, a Sequence Diagram (a combination of flowchart and state chart) to help foster non-linear interactions; and Timelines for specifying and understanding time-based interactions. We conducted research on how best to implement these representations as they are all commonly used in software applications.

When we had implemented the Triggerset interface and the representations, we conducted an evaluation to assess how well users could understand and work with them. We also wanted to investigate the areas that scaffolding would best address. We therefore conducted an exploratory study, which focussed on analysis of VR interactions. Participants were divided into two groups, one of which received additional representations and one of which did not. The study had two parts, in both of which participants were required to examine existing Triggersets without viewing them dynamically in a 3D window. In the first part, participants found possible sequences of interactions; in the second, they found errors in Triggersets. Participants were observed and interviewed afterwards. The results from this study confirmed that the Triggerset formalism was easily understandable; all participants were able to understand individual Triggersets and complete some portion of the tasks. We also confirmed that the representations provided a definite advantage in analysing the design of actions in a VE. In both the sequencing and debugging activities, those with representations performed much better than those without. Some important changes and additions to VRBridge came out of the observations and interviews.

- Areas where scaffolding was necessary were indicated, the primary ones being 3D movement and rotation. Others were the 3D coordinate system and likely debugging options for interactions. A slight surprise was that the representations (which had been viewed as a form of support themselves) required more scaffolding so that their full usefulness could be made apparent to users.
- More interconnection was needed so that users remembered to use the available facilities. The Timelines were barely used because they were not accessible enough. Therefore we needed to increase accessibility and visibility of VRBridge components.
- Users sometimes did not make valuable conceptual connections between the representations and the Triggersets, which prevented them from using VRBridge effectively. Adding dynamic linking would increase conceptual understanding of the relationships between the representations. Therefore, we designed the *Run Mode*, which traces the interactions in all of the representations and on the Triggerset interface as they occur in the 3D world when a player is interacting with the VE.

After this study, we completed the design and implementation of VRBridge. We added the 3D window, dynamic linking, the Run Mode and the scaffolding. We designed three kinds of scaffolding:

- Tooltips for explaining terminology and how to work with the VRBridge and the representations;
- Context Sensitive Hints for encoding domain knowledge and explaining concepts, including assisting in typical debugging problems; and
- Wizards for addressing more difficult parts of 3D interaction design, such as 3D transforms.

1.3.4. Final Evaluation

Our third and final study was a large scale experiment designed to evaluate the relative advantages of additional representations and scaffolding in helping end-users to create interactions, in terms of completion

of tasks, effective exploration of the tool and domain, satisfaction and learning. We hypothesised that both would improve all these areas, but that additional representations would improve task performance, exploration and satisfaction more; and that scaffolding would improve learning more. We also wanted to evaluate the following: how participants used the VRBridge over time; how they used the scaffolding over time; and the differences in usage of the various representations and scaffolding forms.

We created a 2x2 between-subjects and 1x2 within-subject repeated measures design. Participants used VRBridge on two occasions a week apart, and were given three tasks to perform. Each task involved creating a set of interactions involving the player. These were clearly specified to allow for creativity, but to minimise the effect of individual creative differences on the results. We chose three forms of measurement, so that we could triangulate our results and gain a more complete evaluation: task performance, including the extent to which participants explored VRBridge; questionnaires; and logging files of VRBridge usage. Our task performance metric provides a novel way of evaluating the domain expertise with which the task is perform and the tool is explored.

In general, all participants performed better and enjoyed the experience more in the second visit, showing the success of VRBridge. Participants with additional representations or scaffolding performed significantly better than those without, showing the success of our design aids. Additional representations improved exploration and satisfaction more than scaffolding, but not task performance; scaffolding did not improve learning more than representations, but the combination improved learning and satisfaction with learning more than either alone. These results provided us with evidence of the success of our constructivist design process and how to support end-users in authoring VR interactions. It also gave us valuable insights into how we might best support end-users in designing and producing creative artefacts and how we might design effective interactive environments in general.

1.4. Chapter Outline

Chapter 2: Constructivism and Design

We describe the constructivist theory, showing its usage in design. Then we define our design method and discuss our choice of constructivism as grounding theory. We present the five constructivist principles on which we will focus and describe the two features which we will implement: multiple interactive synchronised representations and multiple forms of dynamic scaffolding.

Chapter 3: Related Research on Representations and Scaffolding

This chapter presents a discussion of external representations and scaffolding, focussing on how they can be used in problem solving and design.

Chapter 4: Related Research on Design, VR and Supporting Novices

Here, we present our theoretical research on the development of expertise and support for novices. We discuss how expert designers work, examining studies and guidelines from various design fields. We also present a discussion on the nature of VR, including guidelines on how best to design in VR.

Chapter 5: Classroom Observation of Novice VR Authors

This chapter presents our first study, an observation of students designing 3D games. This study gave us valuable first-hand experience of the difficulties that novices face in authoring VEs, particularly in terms of creating non-linear VEs, with multiple paths for the player to explore.

Chapter 6: Design of VRBridge

In this chapter we describe our initial design of VRBridge, our new VR interaction authoring tool. We describe the modular system architecture and implementation details. We also define the programming style which we chose for our users, including the novel interface for specifying interactions. Then, we describe the design and implementation of our additional representations: Floorplan, Sequence Diagram and Timelines.

Chapter 7: Evaluation of VRBridge

Here, we present our second study, which confirmed the usability of the authoring interface and the effectiveness of the additional representations for improving understanding and debugging of interactions. We also observed ways of improving user interaction with VRBridge.

Chapter 8: VRBridge: Design of 3D Window and Scaffolding

This chapter describes the design of the 3D Window, addition of dynamic linking between parts of the interface and the design and implementation of our three forms of scaffolding: Tooltips, Context Sensitive Hints and Wizards.

Chapter 9: Evaluation of VRBridge: Representations and Scaffolding

This chapter describes our final evaluation. This study evaluated the relative effectiveness and usage of the representations and scaffolding, as well as the usefulness of VRBridge as an authoring tool. The study provided evidence of the usefulness and effectiveness of VRBridge as a basic tool. It also confirmed many of our hypotheses about the effectiveness of the representations and scaffolding, while providing valuable insights into their differences.

Chapter 10: Reflections on VR Authoring, Interaction Design and End-User Creativity

Here, we reflect on the usefulness of our new constructivist design method for VR interaction authoring. We discuss the relative advantages of domain-related representations and scaffolding. We also discuss the implications and generalisability of this work to interactive design systems and end-user systems in general.

Chapter 11: Conclusions

This chapter provides concluding remarks and ideas for future work.

2. Constructivism and Design

When we began working on a project to support end-users in design, we examined educational learning theories and theories of expertise for an appropriate grounding to our research. Constructivism is one of the predominant learning theories (Fosnot 1996), especially in design education (Schon 1987). It is an epistemological theory that describes how knowledge is constructed by the individual through interaction with the world: knowledge is constructed by creating a coherent network of viable explanations for one's experiences. The theory directly examines the complexity of knowledge-making and how it can be managed.

The two other major learning paradigms of behaviourism and cognitivism focus on response to stimuli and internal working of the mind respectively. Constructivism focuses on the interaction between the individual and the environment. Our target audience, end-users, actively become interested in new tools and skills (rather than responding to instruction) and begin learning by playing with software. These details make constructivism the more appropriate choice. We also wanted a theory that was not purely used in education, as end-users are not learning in a traditional classroom environment and must be supported purely through software. Constructivism is widely used in many domains including design and computer usability, as described in Section 2.1, and therefore has broad applicability.

Therefore, we chose constructivism to ground our design process because of its explanatory power and flexibility. Our aim is to investigate how we can best support end-users in producing creative work with an interaction authoring tool; this support means helping the end-user to use the tool effectively, overcome the technical barriers to implementation, and create a good design. Much of this help will involve facilitating learning. Constructivism provides valuable insights into how end-users learn and how this can be enhanced.

Limitations of constructivism are that it is a relativist theory and therefore does not lead to concrete knowledge being developed in the learner; and that, in supporting learning by doing, it hinders the learner from building a deep and consistent understanding of the theory behind activities. While we must be aware of these limitations (and the principles that we distil from constructivism in Section 2.4 are carefully defined to address these), our field of interest requires some flexibility in understanding which makes these limitations less negative than they might be in other domains. Design is an ill-structured problem space with ill defined goals and constraints; learning expertise requires experiencing various possible and competing solutions to problems, and selecting those that work best in the context.

In this chapter we briefly describe constructivist theory and its practical applications. Then we examine how it has been used in design. This provides some background for our theory-based design method, which is described next. Following this, we extract the major principles of constructivism around which we will

structure our design. Finally, we identify the design aids and authoring interface which we will create. These design aids will be the primary means of evaluating the effectiveness of our method and tool.

2.1. Constructivist Theory and its Application

Constructivism is rooted in cognitive science, particularly the psychological theories of Jean Piaget and Lev Vygotsky (Fosnot 1996, von Glasersfeld 1995, 1996). Each theorist proposed a similar understanding of cognitive development; the differences form the basis for versions of constructivism.

Piaget's theory of knowledge acquisition stated that, as new information is assimilated, patterns are constructed, and reflection on these patterns brings structural change in memory. These memory structures are cognitive systems within each individual, with three main properties: *wholeness*, which means that the whole is larger than the sum of its parts, which are interacting and related; *transformation*, which describes the relationships and interactions between parts, how one part is connected to another and both are modified; and *self-regulation*, which means that each structure in memory aims to be organised coherently (Piaget 1937). Vygotsky's theory of cognitive development focussed on social interaction. Information provided through culture and interactions with others is internalized by the individual. Learning is both top down (imposed logic, structure and order) and bottom up (reflection and exploration of everyday experience); and these two patterns are interconnected. Thus, people learn to organise the information that they have gathered, according to directions of others (Vygotsky 1934, 1978).

There are many variations of constructivism which emphasise different constructive forces, e.g., social relationships (Shotter 1995), political and cultural forces (Glasersfeld 1995, 1996), and creating artefacts in the world (Papert 1991). However, they have important similarities which form the core of constructivism:

- focus on learning through activity which is creative and reflective;
- the value of coherence and viability of knowledge above the idea of an abstract truth; and
- focus on interaction within learners between existing knowledge structures and new information, and between learners and their environment.

Constructivism has been applied to many domains, e.g., education, design and psychology. In recent years, it has been applied practically to education and educational software. The cognitive psychologist, Jerome Bruner, could be said to have begun the application of constructivism in education with his work, *Toward a Theory of Instruction* (1966). He also introduced the concept of *instructional scaffolding* (Wood et al. 1974). This was a metaphor for the assistance provided by a teacher which allows learners to perform tasks that are beyond them, and which will fall away when a learner is able to perform the task alone. Another example of the application of constructivism to learning is the use of hypertext systems to guide learners through a

subject, but allow them to forge their own paths and actively construct their own knowledge (e.g., Spiro et al. 1992, Leão 2002).

We believe that the principles of constructivism which support learning are applicable in a much wider sphere than education. People learn constantly, not only in formalised educational settings; constructivism can be used to support all of these learning activities. Specifically, we believe that learning interaction design can be based on constructivist approaches, as can learning the interface to a new system: both of these activities are knowledge-building tasks.

2.2. Constructivism in Design and Competing Theories

Constructivism as it is understood in the design world is largely a theory of visual perception, which suggests that perception involves the internal processing of external stimuli in order to understand them. A major modern contributor is Gregory (1997, Gordon 2004), who proposes that perception is about hypothesis formation and testing. Sensory signals interact with internal knowledge which enables us to make sense of our perceptions. Another major theorist was Rock (1997, Norman 2002), who considered perception to be logical and inferential, much like problem solving, and involving many of the same cognitive processes. This is considered a top-down approach to perception.

The competing theory in perception and design is the ecological approach of Gibson (1979, Gordon 2004), which states that perception is direct and involves none of the internal processing required by constructivists. This is a bottom-up approach to perception. The two approaches are generally seen as incompatible; however, there are many points of connection and attempts have been made to combine them, e.g., the dual-process approach of Norman (2002, Neisser 1995). An example of a connection between the theories centres on affordances. Gibson defines these as perceived invariants of the environment which signify allowable actions; they require no additional internal processing as we have inbuilt perceptual mechanisms which automatically recognise them. However, when discussing pictures, Gibson states that:

"Any picture, then, represents what its creator has noticed and considers worth noticing. Even when she paints a fiction or fantasy, she does it with invariants that have been noticed in the course of learning to perceive." (Gibson 1971, p. 274).

This suggests that Gibson acknowledges a process of learning to perceive. Our premise is that, while this process may occur early and naturally for our perceptions of the world, it is not the case with representations and design work. While expert designers and VE authors are able to perceive the affordances in the environment and objects of a 3D world that allow them to create new designs, novice designers do not know

how to perceive them. These affordances are not the same as those in the natural world. They have been created along with the representations which include them; therefore, they are not necessarily perceived by the naïve observer (Norman 1998). Constructivism can be used to highlight the affordances in representations and help novices learn how to perceive information. We believe that novices can learn through this process to perceive representational affordances as directly and automatically as experts.

Constructivism in design is also about fostering multiple perceptions and interpretations of the designed product: specifying usability without controlling its use and providing a space for interpretation. There are many exciting design practices which have followed these goals in recent years (e.g., Sengers and Gaver 2006). Our position, while maintaining the above values, is guided by the fact that we must help novices with VE design by providing constraints which can later be dispensed with. Without external guidelines, novices are handicapped by their lack of knowledge about what is possible in a completely open design space; therefore, we must constrain interpretation to some extent, while still maintaining flexibility (Laurel 1993). Our aim is to encapsulate the expert knowledge that novices lack in a way that is easy to learn, while making the creative design space as open as possible.

2.3. The Constructivist Design Method

A theory-based design method offers the advantage that the principles and insights of the theory can facilitate the creation of a design based on thought-out and coherent values with understood effects. The first step is choosing a theory; however, the design process must also be structured around the theory to support its continued influence. Having chosen constructivism to guide our research, we developed a novel four-step design method to help us follow the theory consistently throughout our process (Winterbottom and Blake 2004, Winterbottom and Blake 2008).

Our theory-based design method is situated within the paradigm of user-centred design (Abras 2004), as it focuses on the needs of both the end-user of the authoring tool (the designer) and on helping the designer to create an exciting product for the player (the user of the designer's VE). It departs from this paradigm, in that the users of the design tool may not be experienced in the field of design, the domain or programming. As well as learning about a new tool, the user is often learning about how to author within a new domain. Therefore, cycles that only focus on the needs of the experienced user are of limited use; novices are less likely to be aware of the tool features and task support that they will need. This problem is common to many tools which aim to support end-users. The typical user-centred design method is more suited for tools and products that have well-established task structures, where the users are experienced in the domain, e.g., accounting or word processing packages.

Each step of our method has similarities to other design methods, but its novelty is the focus on a single over-arching theory which structures the design, and the focus on examining the processes of both experts and novices. The steps of our method are as follows:

1. *Distil practical values from the theory to guide design and consider how to apply them in practice. Understand the potential implications and impact of these design values*. This step, combined with our choice of constructivism, is similar to the model for designing Constructivist Learning Environments proposed by Jonassen (1999). However, his focus is on using constructivist principles to guide learners in solving a learning problem and instructors in assisting them, not on creating tools for end-users to accomplish practical work without human assistance.
2. *Examine processes of expert designers and experts in the domain and connect them to the design values.* Typical design methods suggest using expert advice in system building, but there is generally no consideration of domain experts, except where they are the users of such systems. Methods tend to advocate using expert design guidelines before implementation, e.g., Norman (1988) and Shneiderman (1992), and expert examinations of the interface after implementation, e.g. cognitive walkthroughs and heuristic analysis (Preece et al. 2002, Nielsen et al. 1990).
3. *Conduct research into the practices of novices in the domain area to understand their specific needs and problems.* This step again has similarities to user-centred design (Preece et al 2002), as we conduct user studies to discover their needs. However, we work on the assumption that end-users will be working in a domain with which they are unfamiliar, and our studies are conducted before a prototype is designed.
4. *Create a prototype and show it to the target audience. Let them use it. Focus on the design values and their effects in questions afterwards. Then incorporate what has been learned into the artefact. Repeat this step until the goal is met.* This step is similar to both adaptive iterative software engineering approaches, such as Agile software development methodologies[1] and user- centred design approaches. It specifies iterative changes to the prototype based on end-user experiences. However, it does not have the process-driven rigidity of these approaches, e.g., there is no time scale or team size prescribed as different design projects will have different requirements in this regard. Among the Agile programming methodologies, it is most similar to Extreme Programming, especially as discussed by Mirakhorli et al. (2008), who advocate customisation of the method for different projects.

As can be seen, our method is relatively generic, without prescribed steps for implementation, which makes it applicable across design domains with different requirements. The novelty of our method is not in its individual steps, but in its overall focus on end-users who are novices in programming, the domain and

[1] http://agilemanifesto.org/

design; and its ability to be applied across design domains. Below we describe the genesis of each step in our method.

Step 1

On deciding to use constructivism as a base for the development of an authoring tool, we extracted usable principles from it for two reasons. First, we believe that for any theory to be used, it must be made practical by distilling useful principles of action. Theories are by their nature abstract, as they involve thinking about higher-level reasons for how things and people behave. To use these ideas practically requires understanding how the theory can be made concrete. Second, constructivism is an especially broad and complex theory. It has many variations, often depending on the field to which it is applied, but they all connect to the same body of thought. In addition, constructivists propose various primary constructive forces, as mentioned above. The different ways in which constructivism is framed make it necessary to uncover the core values of the theory that can usefully be applied to a particular domain. Before further research is conducted, it is important to define these values and understand how they can be practically applied.

Step 2

The next step is examining general expert design practices, and the practices of experts in the domain (VR interaction authoring in this case) in the light of our constructivist values. Expert design practices in any field can provide general design guidelines. Design has been defined as an ill-structured problem space, with no predefined goals or constraints (Dorst 2003, Simon 1973, Jonassen 1999, 2000). This view means that creating a design is a problem-solving exercise, which is difficult to do without constraints, especially for the novice (Laurel 1993). Experts have constraints built in because of their experience and knowledge. A design-aid can help novices by providing the constraints and goals which they lack, with expert knowledge built into the tool and support informed by domain specific guidelines. General expert design practices, like user interface design guidelines (e.g., Norman 1988, Shneiderman 1999), are useful as a starting point for dealing with common problems. For specific goals and constraints, we turn to the practices of domain experts. It is important to connect expert knowledge to the theory-based design values, mostly because there are a variety of guidelines and best practices. They cannot all be implemented, and the design values guide the selection and combination of a set of elements.

Step 3

After research into expert practices, research is conducted into novice practices both in general settings and in the domain. General research on how novices interact with new ideas and how end-users interact with new software can provide initial ideas about how to provide support. Conducting studies to examine how novices work in the domain provides first-hand knowledge about end-user needs and problems. Even if the product is not designed to be used by domain novices, examining novice practices is still useful, as end-users of the

product will be novices in its interfaces and will need to be supported in learning these. Guidelines from this research must be connected to the design values for the same reason as given above: the values provide focal points for the information and therefore guide its selection and use.

Step 4

The final step brings together the previous steps in an initial prototype. This must be shown to the end-users for whom the product has been designed to confirm that their needs have been met, and to find areas of improvement. Information can also be gained on how to assist end-users in working with the interfaces of the product. Ideally, both subjective and objective feedback (i.e., interviews/questionnaires and quality of output) should be gained, so that the designer can understand how well the product assists creative work in the domain, and how the participants feel about working with it. Without some degree of satisfaction, end-users are unlikely to continue using a product. At this point, the design values should be considered to confirm that they have been followed, and so that they themselves can be evaluated for effectiveness. Therefore, this step provides evidence about the success of the product, the success of the design method, and the successful implementation of the theory. The knowledge gained can then be applied to the prototype. This step should be repeated until the needs of the end-user are properly met.

Our theory-based design method allows one to take a theory and practically apply it to a product. An idea for a design is only as good as its implementation, as this is what the end-user experiences. Our design method makes communication between design partners easier, because the principles of design are made explicit as are influences from experts and novices. In addition, by creating prototypes in an iterative cycle, all design partners can view the product of the design before it is finalised and examine the effects of its use. We believe that our theory-based design method has general applicability, as it is flexible while creating a structure in which a design can emerge. Design can be applied to anything and each domain brings its own difficulties. Therefore, there must be freedom built into any generic methodology, in order for it to be effective across domains. From the perspective of creativity, it is also preferable to provide the designer with as much flexibility as possible. Our theory-based design method specifically lays out the steps to access domain knowledge and provides structure for it to be used in design tools. It makes space for domain-specific investigations while remaining relevant across domains.

2.4. Five Principles Derived from Constructivism

Once we had decided to use constructivism as the theory grounding our design, we derived practical principles within it that we could use. This section corresponds to Step 1 of our theory-based design method. Through careful comparison of different constructivist accounts (e.g., von Glasersfeld 1995, 1996, Shotter 1992, Vygotsky 1934, 1978, Piaget 1937, Papert 1991), we distilled five distinct values which are described

below. We also discuss how each principle connects with other areas of research and can be applied to interaction design.

2.4.1. Atomic Simplicity

Complexity often arises from the ways in which simpler atomic parts are connected together. For effective knowledge-building in constructivist terms, each part of a whole or new piece of information should be kept as simple as possible. The complexity of knowledge can be built up through the links between these parts. This makes new information easier to assimilate and provides incremental building of knowledge. Atomic simplicity is also one of the guiding principles of hypermedia systems such as wikis, which contain complex information, but each page is designed to be clear, coherent and simple (Leão 2002). This idea is echoed in programming and user interface theory, where simplicity is valued, but complexity should also be possible (Shneiderman 1992).

Design of interactions and VR are both difficult subjects; therefore we must aim to simplify, as far as possible, the interfaces which we present to the end-user, so that learning a new system will not be an additional difficulty. We must also simplify the authoring process as much as we can without decreasing its power to create effective designs.

2.4.2. Multiplicity

Constructivism is a relativist theory as it opposes the idea of an absolute, objective truth (von Glasersfeld 1996). Therefore, it encourages multiple perspectives on concepts and methods of approaching a problem. Just as we construct our knowledge in myriad ways, so we can analyse it from multiple perspectives, with multiple methods. Complexity is broken down and understood by focussing on different aspects of it. This idea corresponds to visualisation theory, which suggests that different views of an entity are appropriate for different kinds of data and tasks (Horn 2001, Shu 1988). When distinct representations are understood to be describing facets of the same idea, a deeper understanding is fostered (Baldonado et al. 2000). Various existing design systems promote the use of multiple representations, which are often juxtaposed to allow users to think in different ways about their creations, e.g., Harada et al. (1996) and Scaife and Rogers (2005).

A design tool can help the end-user to perceive and interpret multiple perspectives: it can provide multiple ways of viewing and understanding the design (which also simplifies communication of domain knowledge); and support the end-user in connecting them into a coherent understanding of the domain.

2.4.3. Active Exploration

In constructivism, learning and constructing knowledge are active tasks. Successful experiences play an important role in our future actions, especially for novices. Therefore, the potential for exploration is valued

in any tool. If we consider hypermedia systems, we see how exploration is essential to an effective system: because the complexity of the system stems from the linkages, these must be easily and coherently navigable so that the system can be used effectively. Part of exploration is making mistakes and learning from them. Therefore, errors can be seen as a mechanism for users to gain insight, as they provide feedback on the design status. This means that they should be easily identified by the user and informative help should be provided in recovering from them so that they do not become learning barriers (Ko et al. 2004). General user interface and HCI design guidelines support this principle (Shneiderman 1992).

By making exploration attractive, a design tool makes it easier to learn both how to use the tool and how to work in the domain. As mentioned above, one way in which this can be done practically is through the way in which errors are handled. If errors are treated as an integral part of the exploration process, they can be used to provide feedback about what is required to complete a design task. Another way to encourage exploration is to make sure that artefacts can be developed quickly, so that the user gains positive feedback on successful work and wants to explore further.

2.4.4. User Control

Allied to the concept of active construction is the idea of personal control. People gain power over their learning processes by actively constructing their own knowledge. In constructivism, users are provided with opportunities and incentives to build knowledge themselves. This connects with the idea of supporting end-user programming and customisation, which gives end-users more power in their creative activities. The concept of *scaffolding* is used to describe this kind of help: guidance is provided in the form of artefacts, advice and tutorials, which allow learners to perform tasks that would normally be beyond their ability, but which fall away when learners have constructed the knowledge and skill to accomplish the task alone. The educator becomes a coach rather than a teacher (Schön 1987). According to user interface theory (Shneiderman 1992), when a user has control, this generates feelings of success and competence, which encourage exploration.

A design tool should provide users with as much freedom of expression as possible. At the same time, it should provide help which can be tailored to users' capabilities, so as to leverage their skills and allow them to accomplish their design goals. This is more difficult than when an educator is available to assess the situation, as a software system is less able to judge the state of each user. However, various levels of guidance can be provided, and end-users can choose the level which suits them.

2.4.5. Reflection

While activity is important in constructivism, the process of constructing knowledge requires acting with reflection so as to build effective connections between bits of knowledge. By reflecting on their

constructions, people can form viable theories about their knowledge and how it fits together. They can then act with this knowledge. Donald Schön (1983, 1987) developed the idea of *reflection-in-action*, where he considered how students learn to become reflective practitioners in a design studio. The students learn through practicing design activities and thinking while doing them. This principle is echoed in software engineering theory, where iterative methods promote reflection so that each iteration improves on the previous one (e.g., Sommerville 1992). In addition, work on external representations indicates that they encourage reflection and therefore promote a deeper understanding of the subject of the representations (Shu 1988, Blackwell 1997, Do et al. 1996, Eastman 2000).

If a design tool promotes reflection about the design process and its results, it is an effective aid to learning and understanding the domain. Therefore, end-users must be supported not only in creating a design and implementing it, but in thinking deeply about the consequences of their actions within the domain. Providing multiple perspectives on the design, and allowing for early prototyping and feedback provide a mechanism for doing this, as end-users contrast the perspectives and evaluate their creations throughout the design process. In this way, the tool can provide mechanisms which allow the designer to step back from construction and think about how the player will experience the creation.

2.5. Application of Constructivist Principles

While each of our constructivist principles can be used practically in designing an interactive tool, we wanted to define specific *design aids* which incorporate them. This would allow us to focus our design and evaluation. It also alleviated the necessity of creating a complete authoring tool for VR interactions, which would require many components; entirely designing, implementing and evaluating every component is too complex a task for the scope of our work. Therefore, we reduced our focus to the authoring interface and two design aids; with these three components, we can support the essential aspects of a complete authoring process. The authoring interface provides the mechanism by which end-users create their interaction designs and realise them. Therefore, it is a critical component of an authoring tool. The design aids, because they are an addition to the basic tool, can be designed flexibly and applied in other domains more easily than fully integrated components. Since we want to investigate how to support end-users in creative authoring generally, this separation is desirable.

During our extraction of constructivist principles, two features emerged which incorporated all of the principles and which were often discussed in constructivist literature: multiple representations and scaffolding. Both features have an extensive history of research in various domains, which provides us with important insights into how to implement them successfully. Below, we discuss the two design aids and the authoring interface.

2.5.1. Multiple Representations

Support from constructivism for using multiple representations as a design aid can be found most directly in the principle of *multiplicity*. By providing multiple representations of the authoring process, we help end-users to focus on important aspects of the design. However, the problem must also be understood as a whole, which means that the representations must be conceptually related to each other to help the end-user make the transformation. By linking, juxtaposing and synchronising the representations, we can help the designer to understand that they all refer to the same problem-space, but have different foci.

In incorporating multiple representations into our tool, we address the other principles of constructivism as well. By providing multiple ways of viewing the design, we encourage *exploration*. This can be increased by allowing manipulation of the representations, so that the end-user can actively explore them. As users are able to interact with the representations flexibly (rather than in an imposed process), they can actively work to understand the elements of the design and how they come together. They have more *control* over the design process than if the tool only showed the final product. By providing multiple ways of thinking about the design and how it fits together, we also support *reflection* in the user. This is aided by synchronising the representations, and making them relate specifically to domain-relevant issues rather than tool-related issues (e.g., how the designed interactions work together as opposed to how the programming of interactions is accomplished within the tool). Finally, by allowing the design to be viewed in various ways, focusing on different aspects, we *simplify* the information that the designer must comprehend.

Using multiple representations to address the principle of *simplicity* requires some justification, as there is evidence that they can be confusing for novices to understand (Cheng et al. 2001, Scaife and Rogers 1996). They also increase the information load on the user and require more screen space (Baldonado et al. 2000). However, the benefits of multiple representations described above justify their use. If simple representations are used, this should alleviate confusion to some extent. There are also methods for addressing the cognitive load (Baldonado et al. 2000). We discuss these further in Section 3.1.4.

In terms of VR interaction design, the representations highlight different facets of a VE and its dynamic elements, thereby providing users with multiple simpler ways of viewing and thinking about their designs before (or while) viewing the 3D world, which shows the end-product in all of its complexity. In addition, by designing the representations to address important issues within the domain, we assist the end-user in learning more effectively about authoring within the domain, e.g., representations can make the non-linear nature of VR more obvious. Having decided to use multiple, interactive, synchronised and domain-related representations as a design aid, we reviewed research on the theory of representations and their usage, so that we could understand how best to design and implement them. This is presented in Chapter 3.

2.5.2. Scaffolding

Scaffolding, as stated above, is the provision of help which allows end-users to perform tasks that would usually be beyond their ability, and which falls away when no longer required. The second part of the definition separates this design aid from our first: one could understand multiple representations as helping the user to perform tasks that would not otherwise be possible; however they do not fall away. They will continue to be useful as the designer becomes more experienced. This distinction highlights the importance of scaffolding. It eases initial learning of the domain and tool, reducing the learning curve. In addition, because the scaffolding is separate from the tool, the tool itself can be made more complex and useful for expert tasks. Therefore, the provision of scaffolding broadens the usefulness of the tool.

Like multiple representations, scaffolding addresses all of the constructivist principles. It provides end-users with *control* of their learning process, by supporting their activities. This is increased if we make the scaffolding user-adaptable and non-invasive, as end-users choose when to use the scaffolding. This should promote feelings of success at the obvious progress. In addition, scaffolding helps end-users to perform tasks, rather than automating them. This activity promotes *exploration*, as it is made easier through the scaffolding. The aim of any help function is to simplify the task at hand, and by providing scaffolding of various parts of the tool including the representations, we address the principle of *atomic simplicity*. We can provide multiple forms of scaffolding in order to address the principle of *multiplicity* and achieve its benefits. In this way, scaffolding can be designed that supports both domain learning and tool usage. Finally, as the scaffolding helps the user to understand the interaction design process, it promotes *reflection* on this process.

In terms of VR interaction design, the scaffolding creates awareness of important issues in this domain and how to address them, e.g., how to include the player in the interactions effectively. It also supports the technical authoring process by helping the end-user to use the tool. Having decided to use multiple, user-adaptable and non-invasive scaffolding as a design aid, we reviewed research on the theory of scaffolding and its usage, so that we could understand how best to design it. Again, this is presented in Chapter 3.

2.5.3. Authoring Interface

Although the authoring interface is not our primary interest in the application of constructivist principles, it is important to consider. It provides the mechanism for end-users to implement their designs and therefore learn by doing. When we examine the constructivist principles, we easily find indications of how to support end-users with an authoring interface.

The interface should be as simple as possible, while allowing complexity to develop. The principle of *atomic simplicity* suggests that the end-user should be able to specify individual, basic interactions very simply and link them together to yield a rich dynamic experience. The context in which interactions take place can also

provide variation and richness. This will make the interface easy to learn and yet capable of producing complex interactions, and therefore an effective product. To address *multiplicity*, we allow the end-user to author flexibly by providing multiple ways to achieve a design; this is fostered by allowing for basic interactions to be created which can be composed in various ways.

Simple, accessible methods of programming interactions promote *exploration* as the end-user can immediately work with the interface. End-users should be allowed to make mistakes, view the consequences and correct the errors themselves. Immediate feedback about the status of the construction (e.g., through early prototyping) is very important in this process, so that errors do not lead to frustration. The ability to explore and the provision of a simple method of programming interactions will provide the end-user with *control* of the authoring process. Finally, if the technical aspects of designing and creating interactions are simple and easy to understand, end-users will have more cognitive resources for *reflection* on their designs. We review research on authoring interfaces, including how they can support design, VR design and the development of expertise in Chapter 4.

2.6. Synthesis

In this chapter we have described constructivism and its usefulness as a theory to support end-user authoring. We have also shown how we apply it at several levels during this research: through our theory-based design method it will assist us in our process of designing a tool for end-user authoring; it underpins the support that we will provide to novice VR designers through its core principles and their application in the design aids and the authoring tool; and it underpins the support that we will provide to novice users of our tool as they learn its interface and functionality. In other words, it informs our design process to create an effective interaction authoring system, which minimises the learning curve and maximises the end-user's potential to create designs.

From briefly examining our constructivist principles, we know that we want to create an authoring tool and design aids that are simple to use yet can create complexity; that provide multiple methods and perspectives for design and authoring; that encourage exploration through ease of use, enjoyment and early feedback; that provide end-users with as much control of their design process as possible; and that encourage reflection and therefore deeper learning of both the tools and the domain of VR interaction authoring. In the next two chapters we review research in the light of our constructivist principles and according to our theory-based design method, so that we can create an effective authoring tool with design aids that promote the development of expertise and allow satisfying and successful creative work by end-users who are novices with the tool and the domain.

3. Related Research on Representations and Scaffolding

"The power of the unaided mind is highly overrated. Without external aids, memory, thought and reasoning are all constrained…The real powers come from devising external aids that enhance cognitive activities." (Norman 1993, p. 1)

In this chapter we critically examine related research on our design aids: representations and scaffolding. Both aids offer multiple forms of support for working on tasks. Here, we discuss the nature of that support and how to design aids based on constructivist values that provide support most effectively.

3.1. External Representations

We begin by examining the nature of external representations, focussing on how they support the user in performing tasks. We use the following definition of external representations by Zhang (1997) in our work:

"*…the knowledge and structure in the environment, as physical symbols, objects, or dimensions…, and as external rules, constraints, or relations embedded in physical configurations…"* (Zhang 1997, p.179)

This statement defines external representations broadly, to include textual as well as visual representations. This definition therefore includes the authoring interface of our tool as a representation, as we will use its structure and the rules to guide end-users in their design work. For our additional representations, which form a separate design aid, we focus on more visual representational forms. These are situated in the field of information visualisation, which refers to the provision of alternative visual representations of data to support its exploration and analysis (Card et al. 1999). Since this chapter examines our design aids, most of our discussion will centre on visual representations.

There are many questions about the value of external representations and related challenges in the field, including their relationship with internal representations, which forms are better for different kinds of problem, and how individuals differ in working with them. The wide range of questions stems from the number of aspects that must be considered when defining them, including the nature of the underlying data, how it is transformed into the representation, the knowledge and capabilities of the user, the task for which the representation is used and the context in which it is used (van Wijk 2005, Shu 1988, Blackwell 1997, Myers 1990). These aspects and their relationships are clearly described by Blackwell and Engelhardt (2002) in a comprehensive review of taxonomies for diagramming systems. Figure 3.1 is adapted from their meta-taxonomy and displays these aspects graphically; our main change is to broaden the representations considered beyond diagrams, as we believe that the issues apply to all external representations.

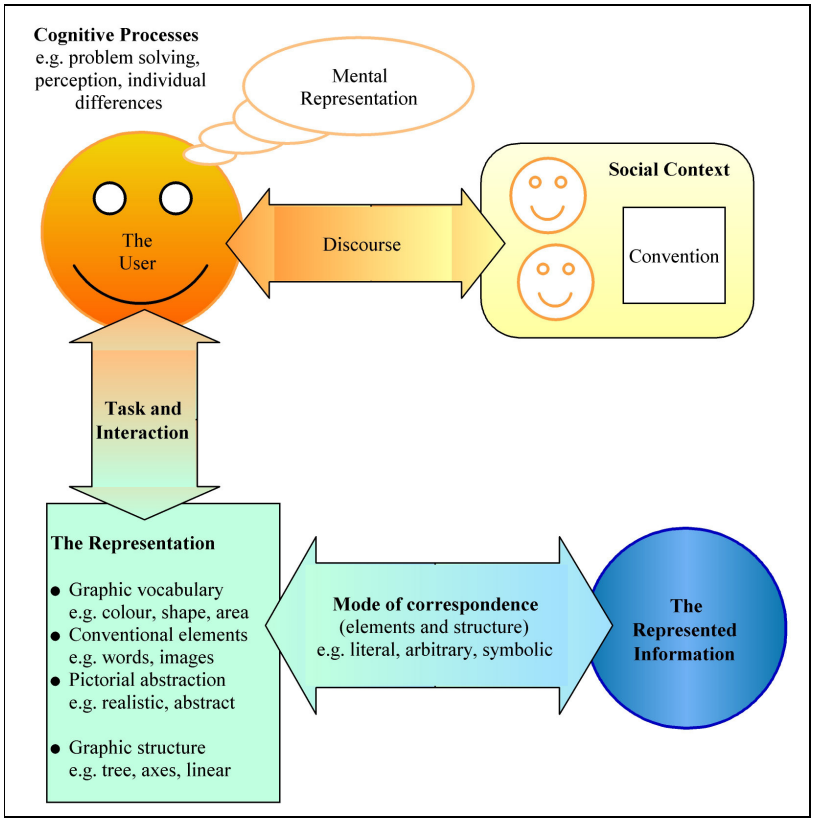


Figure 3.1: Meta-taxonomy adapted from Blackwell and Engelhardt (2002), showing aspects which impact evaluation and understanding of representations.

We have already identified the social context of our representations: the discourses and conventions of the domains of VR and design. Our task is interaction authoring. We shall explore these aspects further in Chapter 4. We discussed the nature of the underlying data in Chapter 1; this will be explored further, along with the specific representations that we use and how they correspond to the data, in our design chapters (Chapters 6 and 8). Here, we focus on the cognitive processes that are supported by external representations and the usefulness of different kinds of representations for different tasks. First, we examine research on the use of external representations to support cognition.

3.1.1. Visual representations and visual perception

Although theorists disagree about the exact nature of the support that visual representations provide, they all agree that visual representations leverage our perceptual abilities. Many studies have been conducted about the human perceptual system and how it works; these provide us with a wealth of information about human perceptual mechanisms (Ware 2000, Card et al. 1999, Rock 1997). The value of perceptual experiments for designing visual interfaces is that we know how to provide the cues that users can easily perceive. Advice includes using the gestalt principles (e.g., connection and proximity) for showing relationships between

elements, using colour and texture perception to design graphics that stand out, and using depth cues for navigation in 3D spaces. Pre-attentive processing of elements occurs before a user pays conscious attention to something, i.e. certain symbols and shapes pop out from their surroundings. Ware (Ware 2000) identifies four categories of pre-attentive elements: form (e.g., line orientation, size, shape and enclosure); colour (e.g., hue and intensity); spatial position (e.g., 2D position and stereoscopic depth); and motion (e.g., flicker and direction of motion).

We will use these perceptual cues during the design of our additional representations. Motion is an especially powerful perceptual technique: it grabs visual attention and highlights patterns in complex data. Moving objects and flashing lights are more easily detected than their static versions as people can see relative motion with great sensitivity. We can lock on to and track objects moving smoothly in the visual field easily using *smooth-pursuit eye movement*. However, if there is too much motion (or other attention-grabbing forms), the benefit is lost as there is too much activity in the visual field for attention to be focussed (Ware 2000, Card et al. 1999). Because of this, we must be careful about how we add motion to our interfaces.

3.1.2. The use of external representations to support cognition

Scaife and Rogers (1996) use the term *external cognition* to describe the cognitive processing of external representations. While the concept of external cognition is useful for considering the cognitive benefits of external representations, it does not encompass the interactive and distributed nature of the process. Ware (2000) describes this as an interactive cycle during which the user constructs and adjusts a conceptual model of the problem, and develops a strategy which contains the representations as key components. Zhang and Norman (1994) state that different external representations of the same abstract task change the way performers work with the task internally; this is the *representational effect*. They argue that in completing a task, a representation is formed that is distributed internally and externally and that both parts are indispensable to task solving. We follow the conception of Zhang (1997), who describes this as *distributed cognition*. He argues that external representations can "*…guide, constrain and even determine cognitive behaviour*" (p.179). The information in external representations is picked up perceptually, but knowledge from internal representations can influence or constrain what is perceived.

Zhang built a framework to describe *distributed cognition*, which is displayed in Figure 3.2. The framework assumes that external representations can be worked with directly as perception is direct; however, they must be perceived so there is need for internal processing. Cognitive operations are needed to direct the perception to task-related elements and perform actions with the results of perception. Each representation is distributed, with internal and external parts, which trigger different operations: external representations trigger perceptual operations; and internal representations trigger cognitive operations. Central control consists of the

mechanisms of attention, working memory, learning, decision making, etc. It coordinates the interaction between perception and cognition, and allocates attention between internal and external representations.

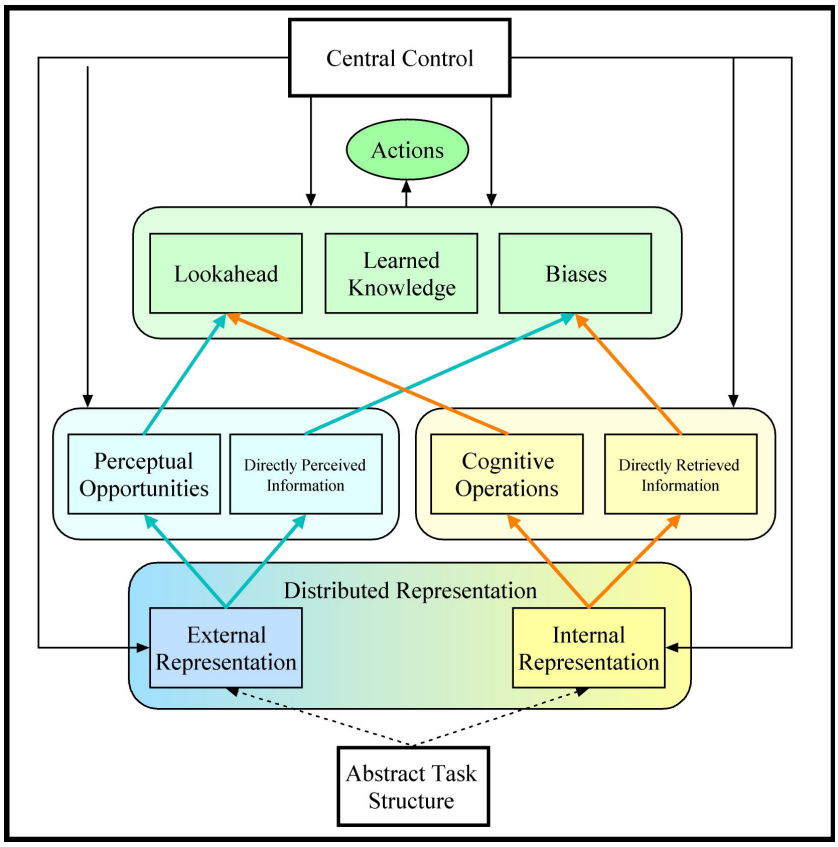


Figure 3.2: Framework for External Representation-Based Problem Solving redrawn from Zhang (1997).

There are three methods of choosing from alternative action possibilities based on these operations: *lookahead*, biases and learned knowledge:

- *Lookahead* is mentally examining and evaluating the results from each action. It is influenced by the complexity of the external problem and limited internal resources.
- *Learned knowledge* results from repeated performance of the task.
- Direct perception and incorrectly learned knowledge can lead to *biases*, which can be consistent or inconsistent with the solution to the problem and so help or hinder the choice.

Zhang applied his framework to an experiment where he created four isomorphic representations of tic-tac-toe (shapes, colours, numbers and lines), and then examined the problem solving behaviour that resulted. The results showed that participants' actions were largely determined by biases from information in the external representations; these were applied regardless of whether they were task-appropriate. Therefore, the form of the representation determined the information that could be perceived and the internal processes activated.

Several theorists have discussed the characteristics of external representations that assist cognition. In Table 3.1, we present some of these characteristics. The results presented in the table are based on research from various theorists, most notably Zhang and Norman (1994), Scaife and Rogers (1996), Cheng et al. (2001), Myers (1990), Ware (2000), Blackwell (1997) and Card et al. (1999).

How External Representations Support Cognition	
Category of Support	**Specific Support**
Computational offloading (reduced cognitive effort)	*Increased working memory* as internal representations do not have to be constructed or are based on external ones
	Direct perception of problem space and its various aspects
	Spatial locality of processing because related information is grouped
	Recognition instead of recall of concepts and terminology
	Pattern recognition through perceptual cues
	Perceptual monitoring of motion or appearance of elements focuses attention appropriately
	Visual organisation of problem space indicates semantic connections
	Increased specificity allows more concrete views of abstract information, improving surface understanding
	Direct manipulation of representations makes consequences of actions immediately apparent
Re-representation (different views of the same abstract data)	*Abstraction* through understanding deep connections between different views of the same underlying data, makes syntax less important
	Relationship and trend recognition through perceiving similarities in data
	Parallel processing of multiple views for efficient search and analysis
	Aggregation and composition through understanding different views as aspects or parts of a larger concept or entity
Constraints (constrain interpretations and inferences about problem)	*Nature of graphical mapping* of elements to data guides interpretations
	Temporal distribution of elements emphasises time-based aspects
	Spatial distribution of elements emphasises spatial aspects of problem
	Spatial structuring explicitly guides understanding of type of connections between data elements

Table 3.1: Ways in which external representations enhance cognition, organised by type of support.

The categories of support are not mutually exclusive and many of the examples of specific support can be placed in multiple categories. For example, pattern recognition through perceptual cues offloads computation

of those patterns as they can be perceived directly; it also constrains interpretation through selection of the perceptual cues to provide. In addition, some of the benefits are in opposition to others because of the varied nature of representations. For example, external representations can increase abstraction or concreteness, depending on the nature of the underlying data. Below, we examine the specific benefits and problems with different kinds of representations, and guidelines for when to use them so that they enhance each other and do not conflict.

3.1.3. Representation types

Cheng et al. (2001) noted that diagrams are difficult to talk about as there are so many different types. In this section we describe common types of representation used for information visualisation, basing our discussion on Shneiderman's seven basic data types (1997): one-dimensional, two-dimensional, three-dimensional, temporal, multi-dimensional, tree and network. We will briefly discuss the advantages of various representation types for supporting cognition, considering constructivist principles. In comparing representations, we can judge whether they are *informationally equivalent*, which means that all of the information that can be inferred from one can also be inferred from the other; we can also evaluate whether they are *computationally equivalent*, which means that all of the information that can be inferred quickly and easily from one can also be inferred quickly and easily from the other. The expressive power of a representation depends on the efficiency with which it can be searched, relevant information recognised and inferences made with that information (Larkin and Simon 1987).

Shneiderman describes text as a **one-dimensional** representation, useful for specifying and finding details (Shneiderman 1997). We do not completely agree with his categorisation, as text documents provide 2D layout cues and can easily be examined non-linearly (Cheng et al. 2001). However, it is useful for separating text from the next category of 2D visual representations, such as maps, so that we can discuss the differences between them. Text is better than visual forms for representing complex, non-spatial instructions, i.e. procedural information, qualifying information, and abstract verbal concepts. In fact, even when diagrams are used, words are often needed to annotate the visual information and provide necessary detail (Ware 2000).

Larkin and Simon (1987) state that visual representations are often more *computationally efficient* than textual ones because the information is displayed spatially and grouped by location rather than connected through symbols: diagrams provide *localization* of elements, *minimization* of labelling and *perceptual enhancement*. Therefore, humans find and recognise information more easily and quickly because of perceptual cues and spatial focussing of attention. Textual representations describe problems implicitly and require mental reformulation (Scaife and Rogers 1996).

Ware (2000) addresses this issue from a different perspective. He distinguishes between sensory and arbitrary symbols. Sensory symbols are those that we can perceive without learning; arbitrary symbols must be learned as they are not based in perceptual mechanisms. The result is that sensory symbols tend to be universal whereas arbitrary symbols are culturally-based. The power of using sensory symbols is that they are easily and immediately understood. The power of using arbitrary symbols is that, once learned, they can support highly expressive notations. In addition, arbitrary symbols can become *overlearned* when they are ubiquitous, and we can use them as easily as sensory symbols, e.g., arabic numerals, natural language. In our representations, we plan to use sensory symbols where possible; otherwise, we will use ubiquitous arbitrary symbols so that they can be easily recognised by novices.

Two-dimensional representations such as maps are useful for showing relationships, especially spatial or structural relationships such as the links between entities. They are also better for location information as stated above (Shneiderman 1997, Ware 2000). Animated images can bring visual representations closer to words in expressive capacity, allowing us to express causality, show transformation and re-arrangement, and sequences of data movements.

Three-dimensional representations such as VR or 3D visualisations are useful for a personal point of view, helping with viewpoint orientation and navigation while viewing data (Shneiderman 1997). Here, we must state that our VR representation is not strictly 3D as we do not make use of a cave system; we use a perspective rendering on screen, which is a 2D representation of three dimensions. However, the benefits of 3D representations mentioned above apply to perspective rendering, as do many of the problems with understanding 3D space. As with one-dimensional representations, using Shneiderman's classification allows us to distinguish the perspective rendering from simple 2D diagrams.

In discussing the effectiveness of using 3D data types for cognition, Scaife and Rogers (1996) question "*…the cognitive benefits of representing the world at higher levels of realism than at higher levels of abstraction*" (p.203). This is a *resemblance fallacy*, as the assumption that people will learn and understand better in 3D is based on its similarity with the world in which we live. Ware also argues that 3D spaces are not necessarily the best design for information visualisation (Ware 2001). This is because human perception of space towards and away from the viewer (z axis) is very different to perception of space to the left and right (y axis) or up and down (x axis). There is far less information available in the z axis. He argues for using 2D layouts to support navigation in 3D space.

This view is supported by many other theorists, e.g., Tufte (1993, 1997), Shu (1988) and Darken and Peterson (2002), as people understand 2D maps very easily. Navigation through new spaces requires building a mental map. This can be done far more quickly and accurately if the whole space can be viewed at

once in an overview. In particular, lines of sight, locations, paths and positions of objects can be easily viewed. People often have problems understanding 3D space and the effects of movement in 3D, and a 2D representation helps to clarify the effects of acting in 3D. It provides a spatial constraint for the placement and movement of objects. Darken and Peterson (2002) reviewed work on spatial orientation and navigation in 3D virtual worlds. They describe the results of several experiments into how to support navigation in 3D:

- Match map orientation to user tasks: a forward-up map is better for navigation and a north-up map is better for planning. Always show user position and view direction on a map and update it dynamically.
- Previous research in urban planning theory has showed that landmarks, routes, districts, edges and nodes are important for navigation. Allow users to annotate maps with spatial cues to create their own landmarks, as this creates a user-defined structure for understanding the space.
- Adding trails to the VE between markers to indicate a path is not useful as the VE becomes cluttered and information will be lost.
- Explicit sectioning provides structure where there is none naturally. Divide big complex spaces into smaller spaces connected in a clear understandable way, and add grids to maps.

Figure 3.3 displays the environment used for several of Darken and Peterson's experiments.

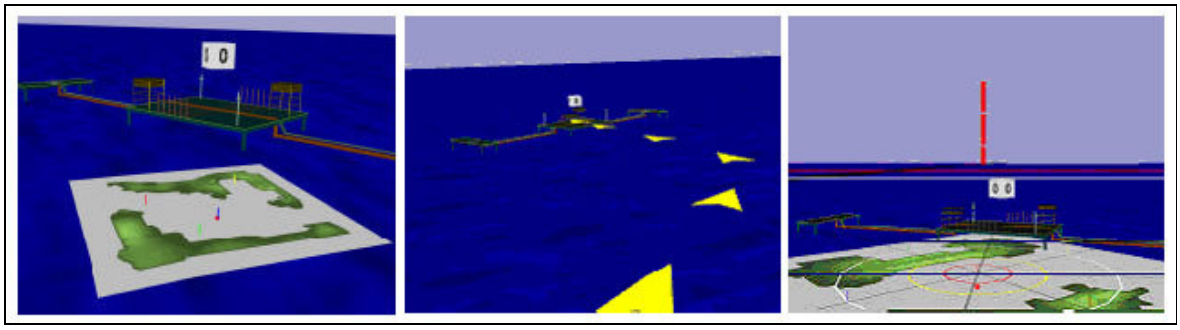


Figure 3.3: VE and maps used by Darken and Petersen (2002) in their experiments: user landmarks on a map and VE; trails in a VE; and radial grid on map and VE.

Tory et al. (2002) conducted experiments that compared the effectiveness of 2D, 3D and combined 2D/3D displays for tasks involving orientation, relative positioning and volume selection. They used three combination views: the orientation icon places 2D and 3D views alongside each other; ExoVis places a 3D view in the centre, which is surrounded by 2D slices; and clip planes display 2D slices of a 3D view in their exact position. 3D displays were found to be useful for approximate navigation and relative positioning when there are appropriate perceptual cues, but are generally not effective when precision is required. Novice subjects found it difficult to work with 2D views alone, but they were useful for resolving ambiguity in the 3D views. The combination displays (especially the orientation icon and ExoVis) performed best for precise

positioning and orientation, both in terms of accuracy and speed of performance, and in terms of subjective user experience.

Temporal representations differ from textual representations, in that they have a start and end time, and items may overlap (Shneiderman 1997). They are useful for finding or sequencing events in time, e.g. before, after or during a specified time period. Timelines provide temporal constraints, as they visualise how elements interact during the passage of time. They are a very well understood formalism and reduce errors in temporal ordering during design (Tufte 1993). Wolber (1998) describes the use of simple, non-branching timelines for the Pavlov Programming by Demonstration system, as this reduces the complexity of each timeline and makes them more computationally efficient for novices.

Multi-dimensional representations such as those used in the visualisation of databases are useful for finding patterns in the data (Shneiderman 1997). However, they are often expert tools, as both the information and visualisation are complex, requiring the ability to read and analyse the data. Because we want to keep our representations simple, we do not consider this data type any further.

Trees and Networks are node-link diagrams used for working with data in hierarchies, for understanding structure and following paths through the data. There are many guidelines for graph drawing aesthetics and guidelines for network diagrams (e.g., Ware et al. 2002, Palmer and Rock 1994, Purchase 1997, Battista et al. 1994, Herman et al. 2000). Typical aesthetics that influence the perception of paths and graph reading are: minimise crossings; minimise area; minimise sum of edge lengths; minimise bends; distribute vertices uniformly; and display symmetry.

State transition diagrams are of interest to us, as they focus on user/computer interactions. They consist of a network of nodes or states and labelled transitions, which make explicit what the user can do at each point and its effects. The problem with most networks is that the complexity of the representation easily becomes unmanageable for large data sets. Harel's statecharts (1988, 1990), which are a form of state transition diagrams, address these issues by creating superstates (collections of smaller states), broadcast transitions to and from multiple states, semantic zooming and other control features.

As our plan is to provide multiple representations of different types, we examine guidelines below for using and designing multiple external representations effectively.

3.1.4. Guidelines for multiple representations

Baldonado et al. (2000) provide guidelines for the use of multiple views in information visualisation. A major consideration is how to link multiple views so that they enhance each other and do not conflict. Two

mechanisms for this are *brushing* and *navigational slaving*. *Brushing* connects the data between views through highlighting: selection in one view results in corresponding highlighting in another. *Navigational slaving* uses interaction to connect views: acting in one view has effects in the others in terms of navigation. Multiple views should be used sparingly as they increase cognitive load through context switching and take up screen space. The authors provide four guidelines for how to use multiple views:

- *Rule of space/time resource optimization* – when deciding whether to present views sequentially or side-by-side, consider the tasks requirements of the user and balance these needs against the spatial and temporal costs of computing and presenting the views. E.g., if the user needs to compare views, the space requirements of side-by-side views are justified.
- *Rule of self-evidence* – use perceptual cues such as coupling or movement to make the relationships between multiple views apparent; this will reduce the cognitive effort required to make connections between views.
- *Rule of consistency* – learning can be made easier if the interfaces of multiple views and their states are consistent where possible.
- *Rule of attention management* – use perceptual cues to focus the user's attention on the right view (or portion of a view) at the right time.

3.1.5. Distributed cognition and constructivism

There is a compelling similarity between constructivist principles of knowledge building and ideas in *distributed cognition*. Both realise the value of multiple forms of representations for improving cognition and the connection between internal knowledge and external information. Both work on supporting knowledge building through active exploration by the user, and promotion of reflection during this process. Table 3.1 and the above sections provide us with some ideas for how we can create representations most effectively to support cognition. Because our potential users are novices in the domain, they are unlikely to have well-formed internal representations. Therefore, the external representations that we provide must help them to consider various aspects of 3D interaction design. We must direct attention to the important parts of representations that will help users to use them appropriately. For any information visualisation task, one should provide an overview, zoom and filtering capabilities and details-on-demand (Shneiderman 1996). This will provide the user with a means for understand the problem space as a whole, and then examining its various aspects.

Canonical forms of representations are important when users are novices, as humans know how to read them across domains, e.g., arrows connecting images in a circle means a cycle (Ware 2000, Scaife and Rogers 1996). If users can recognise a form, they are more likely to be able to use the representation in problem solving. This suggests combining sensory and arbitrary symbols so that in perceiving sensory symbols users

can gain insight into the arbitrary symbols that are provided; and common or ubiquitous arbitrary symbols should be selected. This provides the power of arbitrary symbols while making them more accessible.

Providing interactivity (i.e., allowing manipulation of the representation) helps the user to fully explore a representation externally, rather than having to do it internally. This lessens cognitive load and helps the user to test ideas and gain feedback. This idea conforms to our constructivist principles of *user control* and *exploration*. Ease of production is also important as representation production and comprehension are related; therefore external representations are more cognitively useful if people are able to reproduce them. This connects to our constructivist value of *simplicity* and suggests that we provide simple representations, which users will be able to recreate and internalise more easily (Scaife and Rogers 1996, Eastman 1999).
Combining *multiple* external representations provides support for different kinds of interactivity and helps learners to appreciate abstraction by showing different levels of it, although they must be integrated carefully (Scaife and Rogers 1996, Baldonado 2000). Knowledge stored in memory is more accessible with multiple differentiable cues (Eastman 1999), which are provided with multiple different representations.

3.2. Scaffolding

The term *scaffolding* was first used in the context of learning by Wood, Bruner and Ross in describing one-on-one interactions between a tutor and children of different ages. They described the "…*scaffolding process that enables a child or novice to solve a problem, carry out a task or achieve a goal which would be beyond his unassisted efforts*." (Wood et al. 1976, p.90).

This concept of scaffolding was influenced by Lev Vygotsky's notion of the *zone of proximal development* (ZPD) (Vygotsky 1978). While recognising that what can be taught depends on learner development, he distinguished between *actual developmental level* and *potential development*: the first is the developmental level of the individual when working independently; the second is what the individual can accomplish with the help of others. The difference between these two levels is the ZPD, and support should be aimed at bridging this zone so that the individual can work at the level of potential development. This leads to mental development through *internalisation* of the processes that are learned with the help of others. This concept hints at an important aspect of scaffolding: it must fall away as the learner becomes more experienced, so that the learner performs the task independently and does not rely on the support. This is termed *fading*. We will consider the issue of fading in more detail in Section 3.2.1 below.

We illustrate the ideal process and effects of scaffolding in Figure 3.4, which is adapted from Sherin et al. (2004). In Figure 3.4, the *target performance* is the level at which we want the user to be able to complete a task and provides our criteria for success. The *base situation* is defined by the task, its context, and the

capabilities of the user. The *scaffolded situation* is the same as the base situation, except that scaffolds have been added. This scaffolding improves the *base performance* of the user to the *scaffolded performance*. Over time, the scaffolding falls away or *fades*, making the scaffolded situation more like the base situation. At the same time, *learning* occurs; therefore the base performance without scaffolding improves to be more like the scaffolded performance. Learning also improves the scaffolded performance to be closer to the target performance; eventually, the base performance of the user equals the target performance.

In our case, the task is authoring of interactions within the domain of VR. We will explore other aspects of our particular context for scaffolding during the remainder of this section, while examining pertinent research. Some of the concerns which influence selection of scaffolding are the needs of the user, the source of scaffolding (e.g. computer or instructor), the types of learning supported (e.g. domain knowledge, tool and interface), and the types of scaffolding which can be implemented (Azevado and Hadwin 2005). The discussion will be organised around our constructivist principles.

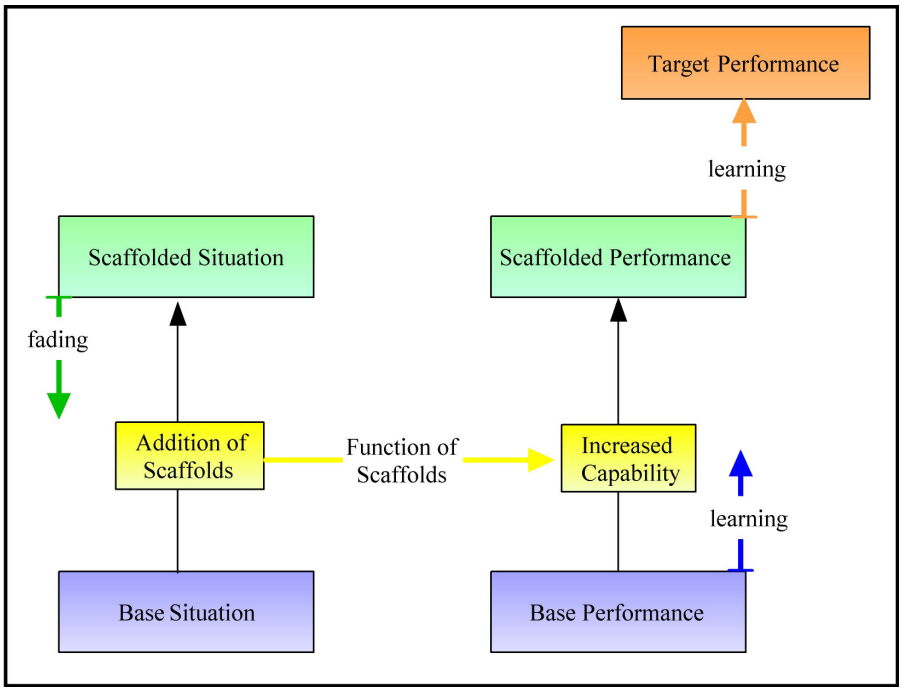


Figure 3.4: Framework adapted from Sherin et al. (2004) for describing how dynamic scaffolding works to improve user performance, showing fading and the influence of learning.

3.2.1. Support versus scaffolding

Some theorists use the term *scaffolding* to mean any support and *dynamic scaffolding* to mean scaffolding which fades (Sherin et al. 2004, Reiser 2004). However, we make a clear distinction between ongoing

support, and support that will fade as learning occurs. While learners can internalise any support and work without it eventually, they are more likely to develop independent expertise if support fades. A study by Kao and Lehman (1997) on learning statistics showed that learners provided with unfading scaffolding did not perform better than those with no support either immediately or a week later. Those who were provided with scaffolding that faded performed slightly better immediately and significantly better after a week. This showed that fading of scaffolding is particularly important for long-term knowledge acquisition.

The difficulties in precisely defining scaffolding as opposed to support are increased when the source of scaffolding is computer-based. Most scaffolding studies involve a classroom situation where an instructor guides the learning process and therefore the usage of scaffolding (e.g., Jonassen 1999, Guzdial 1994, Azevado et al. 2004); this scaffolding will ultimately fade as the learners will leave the classroom. However, when we consider scaffolding with computers, this is not the case. The benefits of computers for thinking and working on problems are well known. Jonassen and Carr (2000) describe computers as mindtools which are themselves learning scaffolds, since they scaffold the way people think by changing the task and providing multiple supports: structural, logical and systematic. However, although the conceptualisation of computers as mindtools is useful in understanding the support that they can provide, we disagree with the authors that this makes them learning scaffolds; the central principle of fading support is not considered.

Jackson et al. (1998) defined three types of scaffolding which we can use to examine the difference between scaffolding and support more precisely:

- Supportive scaffolding offers support and advice for doing the task without changing it.
- Reflective scaffolding offers support for thinking about the task, also without changing it.
- Intrinsic scaffolding involves support that changes the task itself, by reducing complexity and focusing attention.

This classification allows us to consider scaffolding on various levels: supportive and reflective scaffolding map to the more strict definition, which requires fading; intrinsic scaffolding encompasses less strict definitions. Although Jackson et al. strictly defined scaffolding as requiring fading, they were unclear on how intrinsic scaffolding would be faded. Intrinsic scaffolding is likely to be more long-term than the other forms, and less easy to fade. Supportive and reflective scaffolding can be faded without changing the underlying task; however, intrinsic scaffolding requires more effort from the novice as, in order for it to fall away, the underlying task structure must be changed. This means that users are less likely to choose to work without it. The scaffolding classification is made more descriptively powerful with the inclusion of intrinsic scaffolding and the acknowledgement that this form does not fade in the same way as the others. We shall provide all three types of scaffolding in our design aid (see Chapter 8) and evaluate how each is faded.

It is difficult to strike a balance between defining scaffolding to include long-term fading and defining it too broadly so that it loses expressive power. We believe that the way to prevent over-inclusion is to consider whether each potential scaffold promotes and supports learning about a task in a way that can become internalised and lead to the support being faded. With this method, we can clearly define scaffolds and discuss the distinction between scaffolds and other supports. For example, software packages for helping learners to simulate and learn about physics principles are intrinsic scaffolds: learners who are interested enough will move on to more complicated tools and systems eventually, thus fading the support of the initial tool. Conversely, a calculator is not an intrinsic scaffold, even though it changes the task and a novice could learn to perform arithmetic without it: it does not support learning of arithmetic, but automates the task so that the user does not have to think about it. Earlier in this chapter we considered multiple representations and how they can be used within our system to support the end-user. While we hope that the user will internalise the representations to some extent and therefore be able to recreate them, they add value to authoring and analysis even for experts. Scaffolding is support that the expert does not need and would therefore consider trivial.

The above examples highlight an interesting feature of scaffolding. It can fade, but does not necessarily fade. Even supportive and reflective scaffolding can be used in an ongoing fashion if the user does not internalise the support that they provide. Of course, this distinction is only pertinent when one considers user-controlled scaffolding. When scaffolding is controlled by a tutor (human or computer-based), it will automatically be faded. We discuss user control of scaffolding in the following section.

3.2.2. Scaffolding and control

As stated above, most scaffolding studies involve human tutors, who monitor students' work, and structure the scaffolding accordingly. In these cases, the aim is to encourage learner interest in educational topics and make learning with the required tools deeper and more effective. Our interest in scaffolding has a different focus. Our target users are not learners in a school setting; they are end-users who *want* to create an interactive, dynamic 3D world for their own purposes, but who do not have the programming or design skills to use more traditional methods and environments. The reason for engaging in the process and the created product may be work or play-related; the important distinction is that the work is not conducted as part of a general educational program. Therefore, we do not have to consider educational requirements such as following a syllabus. This means that we can provide the user with more control over the scaffolding.

Fading scaffolding that is purely software based can either be adaptive or adaptable, each of which has disadvantages (Jackson et al. 1998). Adaptive scaffolding changes automatically using a model of the learner; however, there is usually no accurate model of each learner so the scaffolding fades according to a general model of the average learner. In an adaptable interface, the scaffolding is faded by the user; however,

learners often do not know when it is appropriate to apply or fade scaffolding. Our requirement of user control means that we cannot use adaptive scaffolding as it provides the user with no control over the scaffolding process. Therefore, we must provide adaptable scaffolding. There are various ways of making scaffolding adaptable by the user, depending on how it appears initially: if the scaffolding is automatically available, the user can choose to ignore it; if it must be opened, the user can choose to stop opening it; and if it can be turned off, the user can choose to do this. Our challenge is to provide scaffolding which the user will want to fade when it is appropriate. Our main approach to this problem distinguishes our scaffolding from our multiple representations: the tool and representations should be simple and clear enough that scaffolding is only required to facilitate initial learning of the tool and domain concepts. Once these are understood, the scaffolding should become trivial to users, so that they fade it themselves.

3.2.3. Multiple functions of scaffolding

In considering the different kinds of learning that scaffolding can support, it is useful to examine Donald Norman's conception of the stages involved in performing an action with a tool, and the *gulfs of execution and evaluation* that can occur (Norman 1988). Norman suggested seven stages involved in performing an action: forming the goal, forming the intention, specifying the action, executing the action, perceiving the system state, interpreting those perceptions and evaluating the outcome. Figure 3.5 presents a graphical representation of Norman's seven stages involved in performing an action, showing the scaffolding types that can support the end-user at each stage.

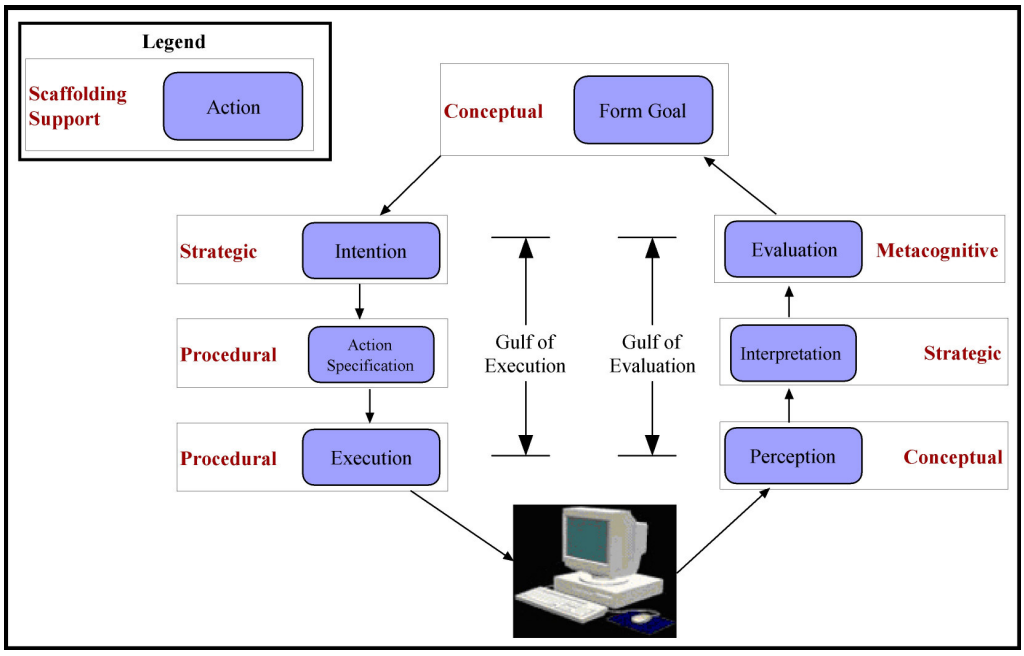


Figure 3.5: Graphical representation of Donald Norman's seven action stages, and the scaffolding support that can be provided at each stage.

The gulf of execution is the difference between the user's intentions and the actions allowed by the system; the gulf of evaluation is the amount of effort that the user has to expend in perceiving and interpreting system state. If the user is a novice working with a new tool, then the differences between user and system are likely to be compounded by a vague and minimal conceptual model of the required task. Our tool must therefore potentially support several kinds of learning: learning a better conceptual model of the task (conceptual); learning how to work with the tool in order to accomplish the task (procedural); learning how apply domain-relevant problem-solving strategies to the task (strategic), and learning about how well one is performing on the task (metacognitive) (Azevado et al. 2004).

Quintana et al. (2002) created a Scaffolding Design Framework to describe the type of scaffolding to apply for various kinds of learning objectives. They identified three general areas of cognitive challenge for learners and associated strategies for support: process management, sense making and articulation. Their framework provides a useful organisation for connecting cognitive challenges and obstacles, and the scaffolding which can address them. In Table 3.2 below, we use the Quintana et al. framework to present a selection of the most typical obstacles facing end-users, the type of learning required and the related form of scaffolding which can be used to address the problem. We have used research by Azevado et al. (2004), Quintana et al. (2002) and Jackson et al. (1998) in this table.

Type of Cognitive Challenge (Quintana et al.)	**Specific Obstacle**	**Type of Learning Required** (Azevado)	**Type of Scaffolding Support** (Jackson et al.)	**Example of Scaffolding**
Process management	Unfamiliar tool and interface	Procedural	Supportive	Hints, Tutorials
	Complex task requirements	Procedural	Intrinsic/ Supportive	Decomposition, Constraints
	Lack of technical skills	Procedural/ Strategic	Intrinsic / Supportive	Automation, Examples
Sense making	Implicit domain strategies	Strategic	Supportive / Reflective	Best practice examples
	Mapping between task and domain practices	Metacognitive	Reflective	Hints, Annotations
	Appropriate focus of work	Strategic / metacognitive	Reflective / Intrinsic	Highlighting, Explanations

Type of Cognitive Challenge (Quintana et al.)	**Specific Obstacle**	**Type of Learning Required** (Azevado)	**Type of Scaffolding Support** (Jackson et al.)	**Example of Scaffolding**
Articulation	Unfamiliar domain terminology	Conceptual	Supportive	Terminology descriptions
	Unfamiliar domain concepts	Conceptual	Supportive / Reflective	Tutorials, Hints

Table 3.2: Table of typical cognitive challenges that face end-users, specifying type of learning required and type of scaffolding to support learning.

3.2.4. Simplification, exploration and reflection

All research on scaffolding follows the definition of providing support during the learning process, and therefore leveraging constructivist principles to enable effective knowledge-building. As can be seen from Table 3.2, scaffolding can *simplify* a task or tool for the end-user; assist the end-user in *exploring* the tool, the task and the domain; and support deeper *reflection* on the meaning of actions in domain terms. Simplification is achieved by techniques such as structuring the task for the end-user, decomposing it into smaller steps or automating technically difficult but less relevant sub-tasks. Exploration is promoted through explanation of strategies and steps, guidance about the connection between task requirements and tool functionality, and support for accomplishing tasks easily. Reflection is promoted through highlighting domain-relevant strategies and concepts, and explanation of domain terminology and how it links to concepts and task steps.

Reiser (2004) argues for a diversity of scaffolding types and approaches to deal with multiple learning obstacles effectively. However, he warns that providing multiple forms of scaffolding which address different problems can introduce tensions. Simplification of work process and task structure can conflict with problematising important content in order to promote reflection. The first reduces cognitive load and makes the task easier for the end-user, while the second adds complexity by introducing conceptual connections which must be considered. Another tension exists between providing end-user control and flexible exploration, and constraining the work process to simplify it.

Our aims with scaffolding are to make our software as easy and intuitive to use as possible, thus reducing the learning curve. We want end-users to be able to create an artefact as quickly as possible, so that they can experience success early and feel able to creatively explore the possibilities of dynamic interaction in a 3D world, without feeling hampered by their lack of skills and experience. However, we also want to support learning about design and authoring of 3D interactions so that expertise in this domain can be developed.

Therefore, as we develop our tool and its scaffolding, we must be aware of the tensions that can be introduced and make sure that the design aids and tool do not conflict with each other or with our overall constructivist principles.

These tensions can be beneficial as they can balance each other. For example, while we reduce the technical difficulties of programming by simplifying the authoring interface, we can promote understanding of underlying programming concepts which impact on the design (e.g., if…then statements). The need to highlight important programming concepts and provide the end-user with flexibility and control should prevent us from over-simplifying the authoring process. Conversely, the need to simplify the complexities of the domain so that they are accessible to novices should force us to focus on the most important domain concepts and strategies. Our distinction between scaffolding and support also serves us in this regard, as each part can complement the others. We can use scaffolding to problematise concepts while making the underlying actions simple to perform with the tool; and we can use it to structure and simplify the working process through explanation and tutorials when the underlying actions must be more complex.

3.3. Synthesis

In this chapter we have examined research on external representations and scaffolding and have shown how both can be used to support the end-user. We have shown how the concept of *distributed cognition* connects with our constructivist values, explaining how we can extend what the end-user can do through effective support and scaffolding.

Jonassen and Carr (2000), in discussing computers as mindtools, divide them into various kinds, which offer different forms of support:

- *Semantic organisation tools* help users to organise and analyse their knowledge, e.g. databases.
- *Dynamic modelling tools* help users to describe dynamic relationships, e.g. spreadsheets and simulation tools.
- *Visualisation tools* help users to reason visually, e.g. information and scientific visualisation systems.
- *Knowledge construction tools* help users to construct artefacts which encompass their knowledge, e.g. tools for designing hypermedia systems and multimedia presentations.
- *Socially shared cognitive tools* help users to collaborate and communicate, e.g. communication tools in large, multi-user worlds.

We hope to support end-users in many of the ways described above. We want to support end-users in creating artefacts containing dynamic interactions between objects in a 3D world, with which they themselves can interact. We will use multiple representations to assist them in visualising the multiple possibilities for interaction, and the constraints on those possibilities. In doing so, they will learn the

terminology and domain concepts necessary to communicate their ideas to others. In combining elements of all these types of systems, we hope to provide increased explanatory and exploratory power to our users as they actively construct their knowledge.

In the next chapter we examine research on supporting novices in general, including ideas on the development of expertise. We also examine the domains of design and VR interaction authoring.

4. Related Research on Supporting Novices in Design and VR

In this chapter we examine research on expertise in design and how to support novices with software in design and VR authoring. Design is a problem solving exercise, with few predefined goals or constraints (Dorst 2006, Simon 1973), which makes it especially difficult for novices. Experts have constraints built in because of their experience and knowledge. A design-aid tool can help novices by providing the basic constraints and goals which they lack, with expert knowledge built into the tool and scaffolding informed by domain specific information. Another tension which stems from the VR domain is that interactions are typically programmed. While we will use alternatives to traditional coding, creation of VR interactions requires logical thinking, e.g. awareness of flow of control. Therefore, the tool must also support learning of these concepts within the domain.

We have described research on supporting novices in Chapter 3, related to how novices use representations and how to provide scaffolding. Here, we focus on advice for supporting novices to design and program effective products. In Section 4.1 we examine research on supporting novices, particularly in design and programming. In Section 4.2 we examine the domain of VR and its specific requirements for expertise and the creation of effective products. Finally, in Section 4.3, we discuss guidelines for supporting novices in authoring VR interactions, based on the research discussed in the previous two sections.

4.1. Expertise and Supporting Novices

Here, we examine the nature of expertise, to discover the general skills which novices lack and how best to support end-users in learning how to design and program interactions effectively. We begin by briefly examining the characteristics of expert work, both in general and in design, and contrasting these with novice characteristics. Then we examine research on how to support novices and experts in working with software.

4.1.1. The nature of expertise

The nature of expertise has been studied in many domains. A foremost theory on the development of expertise is Ericsson's theory of *deliberate practice* (Ericsson and Charness 1994, Ericsson et al. 1993), which states that expertise is gained through long-term focussed, structured and reflective activity in a domain. Deliberate practice is not considered a learning theory, as it focuses on levels of knowledge and skill that go beyond typical learning situations. However, it has powerful similarities to constructivist learning theories discussed in Chapter 2: it requires *activity*, which must be *scaffolded* so that weaknesses can be addressed; and the activity is *reflective*, as it is monitored and feedback provided so that performance improves. Classic findings about the nature of expertise from various fields of study (e.g., Feltovich et al.

2006, Chi 2006, Adelson 1981, Petre and Blackwell 1999, Blackwell 1997, Cheng et al. 2001) show additional connections with constructivism and its usefulness as a method of improving skills:

- Expert performance is characterised by reasoning and the effective use of working memory based on superior knowledge, while novice performance is characterised by general problem solving techniques. An expert's knowledge is organised into a coherent network, which aids reasoning and recall.
- Experts create abstract representations of task features, which are deeper than the surface features used by novices. These representations can be accessed efficiently in a variety of ways and are abstract enough to apply to multiple domain tasks. Novices tend to prefer concrete representations.
- Expertise involves automation of basic operations; as they are practiced they can be accomplished faster and less consciously. Therefore, cognitive resources are more available for reasoning.
- Experts perceive and access information selectively. They detect patterns that convey the abstract structure of a problem. Therefore, only task-relevant information is considered, where novices focus on all information equally.
- Expertise involves metacognition: through reflection and self-monitoring, knowledge is gained about the extent of one's knowledge. Experts reflect on their progress, find errors and backtrack gracefully.

The points above show that experts effectively follow constructivist values: they create and use *multiple* representations or perspectives to perform a task; they actively and *reflectively explore* the appropriate aspects of each task; they can decompose tasks into *simpler* parts by understanding the abstract nature of the problem and how its components fit together; and they have *control* over their working process because they monitor themselves. Our research on design aids in Chapter 3 showed how the provision of scaffolding and multiple representations support learning and task performance; here we see how these aids can work to help novices behave more like experts by increasing the cognitive resources available to them (e.g., through extensions to working memory), providing task-relevant domain knowledge and directing their attention to the most relevant task features. Below we examine descriptions of expert design practices and contrast them with novice work.

4.1.2. How expert and novice designers work

Design theorists characterise the expert knowledge of a designer on multiple levels. The designer has four kinds of knowledge (Eastman 1999):

- factual, e.g. sizes;
- informal, e.g. processes followed;
- procedural, e.g. how to use tools;
- experiential, e.g. knowing what a good design looks or feels like.

This knowledge is gained through the practice of designing and reflecting on that practice, through the materials used and the artefacts created, and through the technical process of implementing the design (Cross

2001, Schön 1987). Scaffolding (which provides information and guidance) and the ability to work effectively within the domain can help novices to develop all four kinds of knowledge.

A seminal discussion of design and how it can be learned is provided by Schön (1987, Schön et al. 1992). He describes design work as a reflective conversation between the designer and his materials: ideas and concepts emerge and are developed as the designer works with the materials required for the task. The designer creates external representations, such as sketches and plans based on initial ideas, examines them and adjusts them continually. Schön describes this as *reflection-in-action*, where the designer notices emerging properties in the materials. Novice designers must be coached while working actively with materials, as reflection-in-action cannot be taught but must be developed through active work. This connects with the *deliberate practice* theory of expertise described above, where expertise is gained through long-term focussed activity in a domain.

Several researchers have conducted studies into how expert designers in various fields create and transform external representations, particularly in how they use sketches (e.g., Do et al. 2001, Kavalki 1999). Sketching allows for multiple interpretations as it is unfinished, which permits more alternative designs to emerge. Drawings are an essential tool in design reasoning; they support a variety of cognitive activities, such as focussing, filtering, remembering, abstraction and evaluation. However, studies comparing sketching behaviour in architectural and engineering experts and novices have found that novices create fewer and less diverse design drawings. Novices also revisit and alter their sketches less, and discover fewer relationships and less deep implications in their sketching (Sobek 2002, Kavalki et al. 1999, Do 2001). In a study by Crismond (2001), designers with differing levels of expertise were asked to redesign tools relying on physics principles. The results indicated that experts recognised and applied underlying principles more than novices; novices only identified important tool features while actually using a device, while experts could infer intended use; and most experts but few novices used drawings to think about the design. Research on the use of external representations in programming has produced similar results (e.g., Shu 1988, Blackwell 1997).

These studies provide practical examples of the theoretical findings on the nature of expertise. They also highlight the importance of effective use of external representations in design work. Novices may not know how to create useful representations or use them effectively for design. However, they can learn from viewing and working with external representations, so that in time these are internalised and become part of the designer's reasoning tools (Eastman 2000, Pane and Myers 1996). They must be scaffolded in this activity so that they learn to use the cognitive resources provided by external representations. Therefore, providing domain specific external representations of a design can act as scaffolding for novices, most effectively if expert knowledge about how to read the representations is included.

Studies have also been conducted into the language and style that non-programmers use to solve problems (e.g., Soloway 1986, Pane et al. 2001). The results of these studies indicate how we can support novices in programming: most novices naturally use an event-based style to specify solutions; novices are often confused by boolean expressions (especially OR and NOT) and avoid creating complex boolean expressions; mathematical operations are expressed in natural language and details are often left out; and novices often use diagrams to describe the layout of the problem solution, but use text to describe events and actions.

Next, we examine research on how to support novices and experts in using software for their design and programming work.

4.1.3. Guidelines for supporting novices and experts in software

Myers and his research group have investigated how to make programming environments more natural for end-users (Myers et al. 2004). One set of studies focussed on *learning barriers* to end-user systems (Ko et al. 2004). Learning barriers are aspects of systems that foster incorrect assumptions about how to proceed. These arise from *simplifying assumptions* made by the user, which may be incorrect as a result of *knowledge breakdowns*. Users will continue to work based on incorrect assumptions as these are difficult to correct once formed (similar to Zhang's (1997) biases fostered by external representations, described in Section 3.1.2). Most types of barriers are integrated with the software itself and occur mostly for novices, such as being unable to find and use functionality, coordinate different parts of a system or understand its effects. However, *design barriers*, related to difficulty in understanding how to proceed with a design, exist for both novices and experts.

Another conception of barriers for end-users focuses on *information gaps* (Kissinger et al. 2006). These consist of: strategy gaps about how to proceed (similar to Ko et al.'s design barriers); features/feedback gaps about the meaning of system features; self-judgement gaps about one's progress and skill; big information gaps where the user is completely uncertain about what has happened; and value/formula gaps about what information to expect or provide to the system. These provide information about the kind of the support that will be most effective for end-users in software.

Blackwell (2001a, 2001c, 2002) considers how to support end-user programming and defines the *attention-investment model*, where the new user of a system will weigh the costs and benefits of overcoming learning barriers to the system, and may decide to abandon the tool if the risks outweigh the rewards. Therefore, systems should aim to minimise the costs to the user (caused by learning barriers and information gaps), while maximising the benefits. This theory is practically supporting by research suggesting that people using a new system will only persevere with problems while they are perceived as relevant to the user's task. In this way, they are less likely to gain expertise, as they prefer to only learn enough about a system to

accomplish the task, and are resistant to the effort required to learn more (e.g., Subrahmaniyan et al. 2008, Ko et al. 2004, Krisler et al. 2008). These studies suggest that the attention required for learning a system should be minimised by providing multiple levels of assistance so that the user can choose how much attention to invest and when to invest it. Learning should be integrated with the task being performed.

Blackwell also suggests several ways that programming systems can reduce costs: represent abstraction so that direct manipulation can be used as much as possible; provide immediate and consistent feedback so that the objects that are being manipulated and the effects of manipulation are always visible in one or another representation; and notations should be easy to read and understand. Blackwell's ideas are supported by studies where designers indicate their ideal tool support and problems with existing software (e.g., Do and Gross 2001, Tangkuampian 2005, Myers et al. 2008). Design experts are often novices at working with computer systems and programming, especially in the field of interaction design where the technological constraints are constantly changing (Dow et al. 2006). The main problems of designers with software tools are: difficulty prototyping through lack of tool support; lack of flexibility in available predefined options for behaviours; time-based interactions are difficult to show with static diagrams and lead to time wasting and loss of accuracy; most tools do not support exploratory design and iteration; feedback is not provided enough and designers have to mentally translate between representations. Designers also have difficulty communicating their interaction designs accurately to programmers when they work in larger teams because of the technical constraints and low-level detail required (Dow et al. 2006, Myers et al. 2008, Tangkuampian 2005). Therefore, common features required for interaction design tools are:

- Support in creating detailed specifications and communicating ideas to programmers.
- Indication of technological constraints on design.
- Support for iteration and prototyping.
- Multiple representations for different views in the design process, linked so changes propagate. The representations need to include support for representing time.
- Support for designer-defined events for specifying interactions, and a flexible work process.

To these specific guidelines for design systems, we can add general guidelines for usable systems (e.g., Shneiderman 1992, Norman 1988, Preece et al. 2002, Nielsen et al. 1990, Tognazzi 2003). The most important guidelines include:

- Make the system status and action alternatives visible.
- Provide immediate and useful feedback.
- Provide a consistent interface.
- Errors should be easy to detect and recover from.
- Provide direct manipulation instead of requiring complex syntax.

In the following section we examine the domain of virtual reality, so that we can define expertise in this domain and understand how to support novices working in it.

4.2. The Domain of Virtual Reality

In this section we first describe the nature of VR, focusing on defining a successful VR application. Thereafter, we review guidelines for designing in VR, particularly designing interactions. During this review, we also examine research from practitioners in the domain of computer game design, as there are important similarities between VR and computer games: both require design expertise and programming expertise; and designers face the difficulty of creating and visualising interactions. In addition, first person 3D games and VR share the difficulty of designing for an independent player whose actions cannot be predetermined.

4.2.1. The nature of VR

"Virtual environments (VEs) allow users to be immersed into three-dimensional digital worlds, surrounding them with tangible objects to be manipulated and venues to be traversed, which they experience from an egocentric perspective." (Stanney 2002, p. xix)

While this definition of VR covers its important aspects, it is somewhat idealistic; most VEs are more impoverished than the definition suggests, both in terms of the degree of immersion and the degree of interaction provided. The degree of immersion can range from wearing a head-mounted display and motion sensors for full immersion in a 3D world, to viewing and interacting with the world using a desktop computer monitor and mouse. The degree of interaction can also range from a walkthrough where the player can only interact by navigating and changing view perspective, to a system where full social and object manipulation are possible. These variations, and the wide variety of VR applications that are used in various domains, make it difficult to extract general metrics for evaluation of VEs. We can begin with the concept of usability from 2D interface design theory: a successful VE must be easy to use and useful, and metrics such as learnability, speed and accuracy of task performance, and subjective satisfaction can be used (Preece et al. 2002). However, these evaluations tend not to be generalisable as equipment and interaction techniques are specialised, and the VR target group is diverse and not well understood (Bowman et al. 2002, Stanney 2002).

Another important concept used to evaluate VEs is *presence*. Presence is generally defined as the degree to which users perceive a mediated experience to be non-mediated (Lombard and Ditton 1997), although there are various definitions. The goal of most VR is to induce a sense of 'being there' in the VE. By investigating presence and how it may be enhanced, general metrics can be developed which can be applied to any VE. Researchers tend to agree that presence is a multi-dimensional concept (Schuemie et al. 2001, IJsselsteijn et al. 2000). In other words, there are multiple possible factors that discourage or encourage presence in the

player. These include naturalness, engagement, and spatial awareness. Several questionnaires have been developed that measure these factors (e.g., Lessiter et al. 2001, Schubert et al. 1999a). However, additional factors such as the form and content of the media which the player experiences, and individual differences in players, also affect presence (Schuemie et al. 2001, Lessiter et al. 2001, IJsselsteijn et al. 2000).

Several theorists in VR have stressed the importance of interactions for creating a compelling world and eliciting presence, particularly the degree to which users can control their actions in the environment (e.g., Schubert et al. 1999b, Carassa et al. 2004, Slater and Usoh 1993, Sheridan 1999). These authors claim that the player's belief in the reality of the world and feeling of presence is largely formed through experimentation with it and interaction with the environment and its objects. If this interaction is rich and fits with the narrative associations of the VE, then the player will feel more involved and more present. Research shows that interaction is not necessary for presence (e.g., Freeman et al. 2000), but that in an interactive system such as a first person 3D world, which suggests the possibility of interaction, presence is enhanced by increased ability to interact (Sheridan 1999). In other words, if users expect to be able to interact (as is the case with VR), increased ability to do so will agree with their expectations and thus improve presence. In the next two sections we examine research on designing interactivity in 3D worlds.

4.2.2. The importance of interaction in 3D worlds

Theorists in games and VR have defined high-level criteria for evaluating 3D interactive environments, which highlight the importance of interaction. Church (1999) defined three "*formal, abstract design tools*" for analysis of computer games: the degree to which games allow player intention to act, provide perceivable consequences and have a coherent story. Murray (1997) defined three aesthetics for creating effective interactive media: agency, immersion and transformation. Fencott (1999) combined the ideas of Church and Murray into four criteria for describing the properties of VEs:

- Agency – the ability to plan goals, perform actions to accomplish them and gain feedback from the VE.
- Narrative potential – the VE is rich and consistent enough for players to construct narrative accounts of their experiences.
- Presence – the sense of being in a non-mediated environment.
- Transformation – the ability to become someone or something else in a VE.

Laurel (1993) defines three axes for characterising the level of interactivity in a system: frequency (how often the user can interact), range (how many choices are available), and significance (how much the choices matter). The first two correspond to Fencott's property of agency, while the third corresponds to creating narrative potential from the actions of the player. Similarly, Salen and Zimmerman (2003) state that in order to create game experiences that are meaningful, players must be able to discern meaningful options for action and consequences (feedback), and action must be integrated into the game context, so that every action is

meaningful in terms of larger goals. To create compelling interactions, they advise designing in iterations and creating simple interactive prototypes with which the designers can play. The following quotation from the designer of games such as *Colonization* and *Alpha Centauri*, illustrates this point:
"…although the individual components look deceptively simple, having a lot of 'simple' moving parts makes for a very complex overall balancing task… To balance all the moving parts correctly, there's no substitute for actually playing your own game—the combinatorial explosion from all the moving parts makes it impossible to truly anticipate or tune results 'on paper' in a design document." (Reynolds, quoted in Saltzman 2003, p. 72)

The above ideas show how we characterise expert design in an interactive 3D world. A critical challenge is allowing player interaction and decision-making while maintaining coherent and meaningful narratives. This considers both interaction and the content of the 3D world (as the content relates to the narrative meaning of the interactions) and how they can be used to create a good experience for the player. There is a similarity between the criteria described above (especially the concept of agency) and guidelines for system design. The guidelines of feedback, visible system status, early prototyping (for a player perspective), good mappings, etc, all work to promote effective agency in the end-user. This suggests a commonality with any interactive system; a primary design focus is the ability of the end-user to interact with the system in an enjoyable and knowledgeable way.

4.2.3. The nature of interactions in 3D worlds

Despite the similarities between system and user interface design guidelines and theories of interaction in 3D worlds, there are differences in the type of interaction. Kaur (1998) developed a theory of interaction in VEs, partially based on Norman's seven stages of action with a system. She defines three models of interaction:

- The *task-based model* describes creating a goal, planning action, performing the action and evaluating it.
- The *explore-navigate model* describes navigation around the environment without a specific goal.
- The *system initiative model* describes reacting to system events.

Bowman and Hodges (1999) created a VR testbed where they explored the effectiveness of various travel (navigation and orientation), selection and manipulation techniques in immersive VR. This taxonomy of interactions fits into Kaur's task-based and explore-navigate models, and the authors claim that system initiative events are made up of the other actions. These models provide us with guidelines for how the player can interact with the VE and its objects. They also allow the designer to decompose interactions and VE design tasks into sub-tasks where interaction combinations can be explored (Bowman et al. 2002).

These taxonomies focus on the actions of the player in the VE. These are the most essential interactions to consider, as the experience of the player is of primary importance. However, interactions that are not

explicitly instigated or performed by the player are also important to the overall experience, as they add to the complexity of the VE world and make it more compelling. This is important for presence, narrative interest and transformation. Bowman and Kaur, because of the focus of their work, only consider the interaction of objects other than the player briefly. For Kaur, they would mainly fall into the *system initiative model* of interaction, which she barely explores. In order to create a dynamic and interesting environment which feels real, the world must seem to exist on its own terms and not only as a space for the player. Otherwise, at its best the experience will be like exploring a ghost town empty of other life.

Wazlawick et al. (2001) provide a taxonomy of interaction in non-immersive VR, which explicitly considers all of the objects in the environment. The authors define six types of interactivity:

- Navigation and orientation;
- Simple reactive objects, which respond to selection by the player;
- Complex reactive objects, which respond to commands from the player with different responses;
- Sensory objects, which react to other events besides explicit player interaction;
- Proactive objects, which may trigger events and actions without player intervention;
- Chatterbots, which are objects that can talk and answer questions.

The authors' conceptualisation is somewhat idiosyncratic, mainly because their system makes use of Chatterbots and therefore they wish to distinguish them from the other types; we believe that Chatterbots are a sub-type of Proactive object. Despite this, the classification provides a useful way to consider levels of interactivity in a VR. If a VR system provides qualitatively different methods of interaction, this will increase the perception of its interactivity. The additional complexity provided by the various methods can also cause unexpected reactions to the player from the VE, which is desirable in terms of creating a compelling experience. With this taxonomy, we can encompass many different kinds of interactivity, while still using only common, basic input devices: the keyboard and mouse. This dynamic content forms part of Fencott's perceptual opportunities and guides the player through the VE narrative.

The abovementioned theorists decompose 3D interaction into different types which correspond to the task of the player. However, Bowman et al. (1999), in their taxonomy of interaction techniques also discuss how interactions can be decomposed to create a flexible system for the designer: "…*we can reduce the space of the problem* (designing interaction techniques for VR) *by recognizing that there are a few basic interaction 'building blocks' that most complex VE interactions are composed of… Since these tasks are so common, the resulting* (interaction techniques) *may be useful in a wide range of VE applications. Thus, the identification and understanding of such 'universal tasks' for VEs is an important first step towards addressing usability difficulties within interactively complex VE applications.*" (Bowman and Hodges 1999, p. 38).

Salen and Zimmerman consider the same process for games:
"*The basic unit out of which interactive meaning is made is the action > outcome unit. These units are the molecules out of which interactive designers…create larger structures of designed action.*" (Salen and Zimmerman 2003, p. 65)

Other theorists have suggested the same concept, both in terms of breaking up tasks into sub-tasks through taxonomies (e.g., Kaur 1998, Zachmann 1996, Bowman et al. 2002), and in terms of breaking up interactions into their basic, simple, atomic actions (e.g., Bowman and Hodges 1999, Tanriverdi et al. 2001). From the above guidelines, we know that VR and game designers advocate iteration, early interactive prototypes, considering player tasks, and breaking interactions into basic building blocks. Many of these points connect with general usability advice for novices and advice for supporting design, as discussed in Section 4.1.

As seen in this section, there are various models for characterising interaction types, and suggestions for important atomic actions to consider. For our work, we have chosen to use Zachmann's (1996) system for conceptualising and structuring our interactions. Zachmann provides a system and set of rules for connecting building blocks of interactions in an event-action system, which is useful when considering how to create such a system. Events and actions are objects themselves, rather than attributes of objects in the VE. The relationship between events, actions and objects in the VE is shown graphically in Figure 4.1, which is adapted from Zachmann.

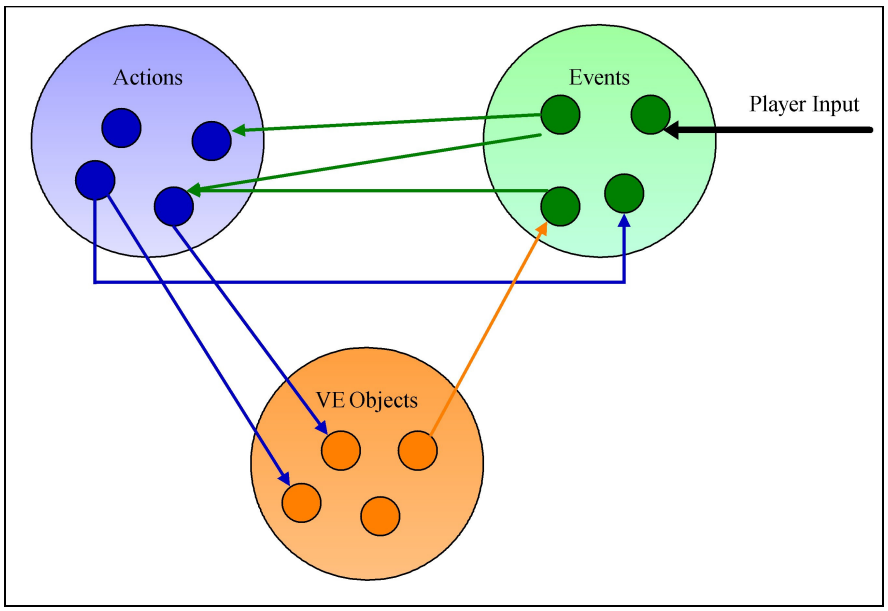


Figure 4.1: The relationship between events, actions and VE objects adapted from Zachmann (1996). Actions are anything that happens in a VE. Events trigger actions, and can be caused by player input, VE objects or actions.

Actions by the player are qualitatively different to other actions. The player independently decides to perform actions within a VE, which may be in response to events, but that are not triggered by events. All other actions are triggered in some way. The player's input is an event in this system. Zachmann states that for an event-action system to be flexible enough to create complex interactive environments, it must obey the following requirements: any action can be triggered by any event; events and actions have a many to many relationship; events can be combined by boolean expressions; events can start or stop an action when a condition holds; and an action can be the input for another event. This provides us with rules for composing interactions from the basic building blocks of actions and events; however, Zachmann does not provide any experimental evidence about the effectiveness of his rules.

This system allows us to specify a range of interaction types. Actions are triggered by events. These are flexibly defined to have three sources: world events, such as the world starting; player input, such as pressing a key or moving into proximity; and behaviour of VE objects, such as playing a sound or performing an animation. This behaviour is itself caused by actions being attached to VE objects. These actions can encompass a variety of behaviours, from starting media such as sounds or animations, to changing properties of the objects, such as their colour and size, to changing position or orientation.

These actions and events can then be composed to elicit complex combinations. For example, when the player turns to look at object A, A starts moving towards object B. When A moves into proximity of B, an action makes A stop and gesture. B then responds. This fairly complex sequence can be created from the simple building blocks of actions and events as defined in Zachmann's system. Uncertainty can also be included to some extent, as action sequence C can be defined as occurring if the player moves one way and action sequence D will occur if the player moves a different way.

Two limitations of this system are: difficulty in dealing with time-based interactions, e.g. wait two minutes and then do action A. These restrictions are considered in Chapter 6, when we introduce our interaction system in more detail. In the next section we explore how to support end-users in creating effective interactions in VR.

4.3. How to assist end-users in authoring interactions in VR

In this section we examine the guidelines discussed previously and distil a useful sub-set for supporting end-users. We divide these into guidelines for system usability, low-level design guidelines and high-level design guidelines, where low-level guidelines address technological issues of working with software, such as programming, and high-level guidelines suggest good design practices. In order to support novices, we must support all three: the authoring system must be usable and easily learnable so that novices will be encouraged

to continue working and learning; novices must be able to accomplish tasks with the system, which means that they need low-level support for programming interactions effectively; and we must support novices in working more expertly within the domain of VR authoring, with high-level guidelines. Before we begin, we critically examine a very successful VR authoring system for novices, *Alice*[2], to understand how this has been done practically. Although we are familiar with several authoring systems, we chose only to evaluate *Alice* because of its suitability and the high degree of research reported by its creators. Most systems are created in industry and do not discuss or justify their choices for interfaces and interaction styles. In addition, we wanted to follow our own design process in creating our authoring tool, so that we could justify the interface and design aid decisions that we were making. Using features from other systems would have hindered and complicated this aim.

Although *Alice* began as a rapid prototyping system for VR which allowed non-technical users to create interactive environments, it changed its focus to learning programming through building 3D worlds. It has been used successfully in many schools and universities to teach and foster an interest in programming (Conway 1997, Conway et al. 1995, Cooper et al. 2003). Figure 4.2 shows the interface of *Alice 2*, the most recent version.

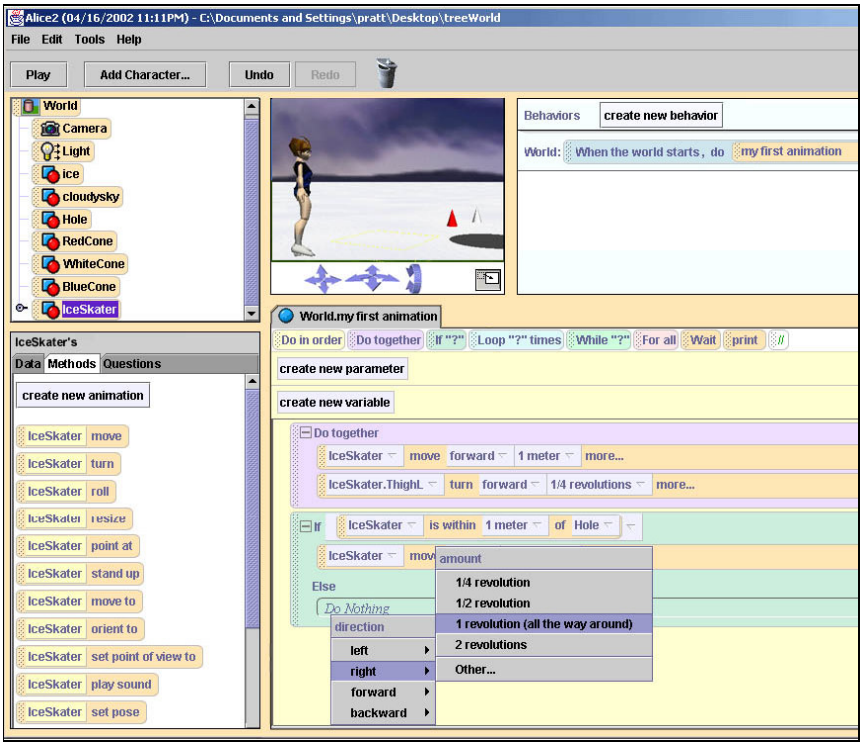


Figure 4.2: Screenshot of Alice 2 showing Methods tab, 3D window, Behaviour window and Animations tab.

[2] www.alice.org

Alice uses an event-based programming style. The system is organised in tabs, with a Screen Editor for laying out the scene and placing objects, a Behaviour Editor for specifying events which trigger animations on objects, an Animation Editor for composing animations (by customising and combining an object's predefined methods), and an Object Editor which contains the details and methods of each object. Extensive user testing was carried out on initial versions, which led to increased direct manipulation and changes in the language for 3D manipulation to make it less technical. For instance, forward, backward, up, down, left and right are used instead of XYZ notation; move and resize instead of translate and scale; and rotations are described as turns per second, instead of degrees per second. Programs can be built incrementally, as each command is tested and animated immediately by the engine, providing immediate feedback.

Direct manipulation is provided with drag and drop functionality, so users can drag objects into the Animation Editor, and select actions and their parameters from menus. These actions are limited, and users must code in Python to create more complex actions. In *Alice 2*, program control structures such as if-statements are available for composing animations (Kelleher et al. 2005). *Alice* addresses technical difficulties by using direct manipulation and constraints on programming actions (as options are selected from drop down lists). Individual properties of objects and actions can be composed into complex interactions.

Alice's content and rules are broad, as the system teaches general programming concepts, but there is no overview of the design or advice about how to create effective interactions. The system provides good usability and learnability, and low-level support for effective programming. However, all higher-level design advice is provided in the classroom or in manuals and not in the tool. Therefore, *Alice* provides effective low-level guidance in authoring 3D interactions, but does not provide high-level guidance in the software.

4.3.1. Providing a usable tool

In Table 4.1, based on our research described in Section 4.1, we provide system design guidelines for creating a usable, easily learnable and enjoyable space for end-users to design VR interactions. We also indicate connections with our *constructivist* values. We include guidelines from Shneiderman (1992), Norman (1988), Preece et al. (2002), Nielsen et al. (1990), Tognazzi (2003), Ko et al. (2004), Kelleher and Pausch (2005), Blackwell (2001b), Subrahmaniyan et al. (2008) and Pane et al. (1996, 2001). As there is much overlap, we do not include individual references.

Guideline	**Details**	**Benefits to Novices**
Visible system status and action	Use techniques like colour, animation and spacing to highlight information; avoid cluttering	Directs attention to important concepts and tool functionality.

Guideline	Details	Benefits to Novices
alternatives	with too many attention-grabbing techniques.	User can explore effectively.
Good mappings and affordances	Obvious mappings between system and domain. Mappings between stages of action, and effects should reveal relationships.	User predicts effects of actions. User connects tool functionality to domain requirements.
Consistent interface	Screens, menus, dialogs, etc. should look similar where similar actions are required. Use familiar conventions and terminology.	User learns system functionality more easily and feels more secure and in control.
Feedback	Feedback should be immediate and appropriate.	User explores system more easily and feels more control.
Flexibility	Provide as much user control and flexibility as possible.	User has freedom of expression. This encourages reflection and exploration.
Constraints	Provide constraints to guide user actions.	Guides users to correct domain actions for more expert usage.
Error handling	Errors should be easy to detect and recover from. Error messages should not use negative or technical terminology and should be precise.	Encourages exploration; assists learning; prevents errors from becoming learning barriers.
Direct manipulation	Provide continuous representation of objects, immediate feedback and reversibility, and actions instead of syntax.	Recognition instead of recall. User feels in control. Promotes exploration.
Multiplicity	Provide multiple ways of encoding and retrieving information, which are dynamically linked.	Helps recall and perception. Helps users make connections.
Scaffolding and additional help	This should be obvious, user controlled, and at multiple levels for long and short explanations.	Provides domain and tool knowledge for more expert usage.
Short development time and prototypes	Allow short iterations and early production of artefacts.	User gains positive feedback for further exploration.

Table 4.1: Guidelines for supporting novices in tool usage within a domain. Shown are guidelines, their details, and the benefits to novices.

The combination of flexibility and constraints introduces an apparent contradiction, which highlights an important tension in any authoring tool for end-users. Laurel (1993) addresses this by suggesting that constraints may enhance freedom of expression and flexible work: "*When a person is asked to 'be creative' with no direction or constraints whatever, the result is … often a sense of powerlessness or even complete*

paralysis ... Limitations – constraints that focus creative efforts – paradoxically increase our imaginative power by reducing the number of possibilities open to us." (Laurel 1993, p. 101) Since end-users may be complete novices in tool usage and domain skills, they will need some direction in the correct actions to take (as indicated by interaction designers in their guidelines from Section 4.1.3). This tension between providing constraints and flexibility ties into those described in Section 3.2 between providing scaffolding which problematises domain content and scaffolding which simplifies work. The challenge is to find the path of support between constraints and flexibility that *focuses creative efforts*.

In this section we have described guidelines for creating a generally useful system. In the next section we discuss how to provide support for accomplishing work specifically within the domain of VR authoring.

4.3.2. Low-level authoring guidelines

Our research in Section 4.1 indicated that an event-based programming style is naturally used by novices when programming, and that direct manipulation is recommended. In addition, theorists in VR suggest breaking interactions into simple building blocks which can be variously composed, which connects with our core principle of atomic simplicity (decomposition of complexity into its basic atomic components). The designers of *Alice* successfully followed these guidelines, and researched user-recognisable terminology that made designing interactions more intuitive for their users. Therefore, we follow these recommendations.

Another question which we must consider is the extent to which we allow direct manipulation in the 3D world. Our interactions will be programmed in an event-based style; however, authoring also requires manipulation of the space in which interactions will happen. Our research in Section 3.1 indicated that novices are often confused by 3D space and that the combination of 2D and 3D representations is most effective for accomplishing tasks.

Additional evidence is provided by Bowman et al.'s research on interaction techniques (2001). They used multiple measures to evaluate interaction techniques for VEs, including speed, accuracy, ease of use and learning, spatial awareness, presence and user comfort. The Virtual Gorilla Exhibit (Allison et al. 1997) was created to evaluate direct manipulation in the 3D world as opposed to using a 2D metaphor. This was an immersive system where student environmental designers could make changes to a zoo exhibit. The authors implemented two interaction techniques: a virtual hand within the VE; and a 2D pen and tablet, represented both physically and within the virtual world, where actions on the tablet are reflected in the VE, e.g. moving the red dot to a new location moves the user there. Figure 4.3 shows the tablet in action. While students could use both interaction techniques in the VE, they tended to use the tablet for moving around the VE to complete tasks and the virtual hand for exploring.

Figure 4.3: Pen and tablet interaction device used in Bowman et al.'s Virtual Gorilla Exhibit (1998), showing both physical and virtual manifestations.

The authors found that students preferred the pen-and-tablet technique to the direct manipulation technique. They hypothesized two reasons for this: the tablet and its icons were always accessible, and provided useful constraints on action (i.e., only two degrees of freedom compared to six); and user familiarity with 2D interaction on desktops made the pen and tablet more familiar. This result connects with our previous research in Chapter 3 about the effectiveness of combination displays. In addition, it connects with general system guidelines about providing constraints for novices working in 3D environments. We have shown how we can support end-users in accomplishing interaction authoring in VR. However, we also want to foster more expert design work. The next section discusses guidelines for how to support this.

4.3.3. High-level authoring guidelines

Feltovich indicates how one might support novices in learning with high-level guidelines:

"*When focusing attention on making a particular decision or solving a particular problem, three types of events occur that are critical for effective reasoning: (1) we seek data from the environment, (2) we bring relevant knowledge to bear from our long term memory to working memory and, by reviewing the data in the presence of relevant knowledge retrieved from long term memory, (3) we draw inferences about what is going on which may lead to seeking more data and activating more knowledge*." (Feltovich et al. 2006, p 58)

We all have limits in attention and perception; we can pay attention to a task for a limited time, have limits in working memory, and limits in long term memory access. Experts combat these limits with selective attention, pattern finding and abstraction, which are gained through experts' knowledge and cognitive representations. If we follow the reasoning steps described above, we see how we can support novices in reasoning more expertly throughout a task: in step (1), critical information in a task can be highlighted so

that the novice perceives it; in step (2), we can provide novices with knowledge and information about how it connects to the task; in step (3) we can assist novices in reasoning with this knowledge and the current task by highlighting domain-relevant inferences and providing external representations which show the process of inference. We discuss how to accomplish these aims within VR authoring below.

Kaur (1998, 1999) identified usability problems for design in VEs and then created a taxonomy of 3D usability guidelines. Many of these are similar to 2D user interface guidelines, which she admits using as a base for her taxonomy. However, she made additions which cater specifically to VEs, such as more emphasis on supporting perception of areas of interest, spatial navigation and viewpoint control. Table 4.2 shows examples from Kaur's guidelines, categorised according to the design issue they address.

Category	**Example Guideline**	**Description**
User task	Clear task/task flow	Task is clearly defined.
Overall VE	Discernible repertoire of opportunities for action	General opportunities for action can be easily determined.
Spatial knowledge	Locatable object/areas of interest	Important objects and areas of interest can be easily located.
Objects	Accessible object	Object can be easily approached and accessed
Actions	Executable action	The action can be executed efficiently and without frequent problems.
Action feedback	Declared action effect/success	Feedback on the effect of the executed action is given, and errors easily detected.

Table 4.2: Examples of Kaur's guidelines for VE design, divided into categories of VE design issues.

Kaur provides forty-six of these guidelines, with examples in each category. Her tests showed that VE users were able to accomplish tasks much more successfully in VEs designed using the guidelines. However, the quantity of guidelines is a great deal for a novice designer to consider while creating a VE. Many of the options overlap, e.g., *clear task flow* and *discernible repertoire of opportunities for action*, and could be covered by "make action possibilities and system state clear", which is the 2D interface guideline.

The research on supporting novices in programming and design suggest that multiple representations should be used with active and dynamic linking to teach conceptual ideas and how to perceive patterns in the representations. These are likely to be more usable for novices than Kaur's textual guidelines, although some of her guidelines could be included in scaffolding. Another reason for using multiple representations to provide high-level design guidance relates to our decomposed interactions. Constructivism states that when

multiple perspectives are used, their interconnection must also be highlighted, so that they can be understood as part of the same concept. Similarly, with interactions, when we break them up into basic actions, it must be clear how to compose them to form complex interactions. In addition, when we are focussing on the typical tasks that the player will perform in a VE, we must consider both the basic, atomic tasks (e.g., move an object) and the higher-level tasks which contain them (e.g., move an object towards another object, avoiding obstacles). In this way, they can understand the *building blocks* of the interaction and the more high-level cognitive tasks which create meaning.

Scaife and Rogers (2005) describe how to foster this cognitive connection when discussing work with new technologies (interactive multimedia, virtual reality and mixed-reality). Users with multiple representations have to translate between them. Therefore, the representations must be dynamically and explicitly linked to each other. In tests with learners, the authors found that learning was most effective when the mappings between representations could be explored, the representations were dynamically linked, and learners could actively manipulate the representations.

Another high-level issue is supporting interaction design in novices that creates a coherent, non-trivial and non-prescriptive narrative. A compelling method is to design the narrative into the space through spatial cues for action, or embedding narrative information in content, e.g. changing scenery to suggest events which have occurred, such as a broken down door (Jenkins 2004, Chen and Kalay 2008). Similarly, Salen and Zimmerman (2003) state that games should provide *contexts for interaction* which players can explore. These can be spaces, objects or behaviours. We can provide design contexts for our end-users which will inspire them to consider narrative in their interactions.

Fencott (1999) describes perceptual modelling as a VR design stage in order to address interaction with content. In order to design for perception of action and narrative possibilities, he defines *perceptual opportunities* of a VE, which leverage perceptual properties of objects in the VE to grab and retain the player's attention; this creates narrative paths through a VE. Perceptual opportunities are composed of sureties, surprises and shocks. Figure 4.4 displays Fencott's classification of perceptual opportunities.

Sureties are familiar objects or patterns in the VE that help the player to believe in its reality, e.g. vection (i.e., moving around in a world), perceptual noise (e.g., dirt), physics, and sounds. These are necessary for VR design because it is not possible to recreate an entire world. Shocks are undesirable VE aspects that break presence, e.g. polygon leaks. Surprises are aspects of the VE that guide the player to narrative experiences. They are divided into attractors, connectors and rewards. Attractors attract the player's attention to areas of interest through visual interest, spatialised sound, etc. Connectors guide the player towards

rewards. They can persuade the player to follow a course (e.g., paths or signs), choose between courses, or avoid a course (e.g., degraded content). Rewards repay players for their interest with memorable experiences.

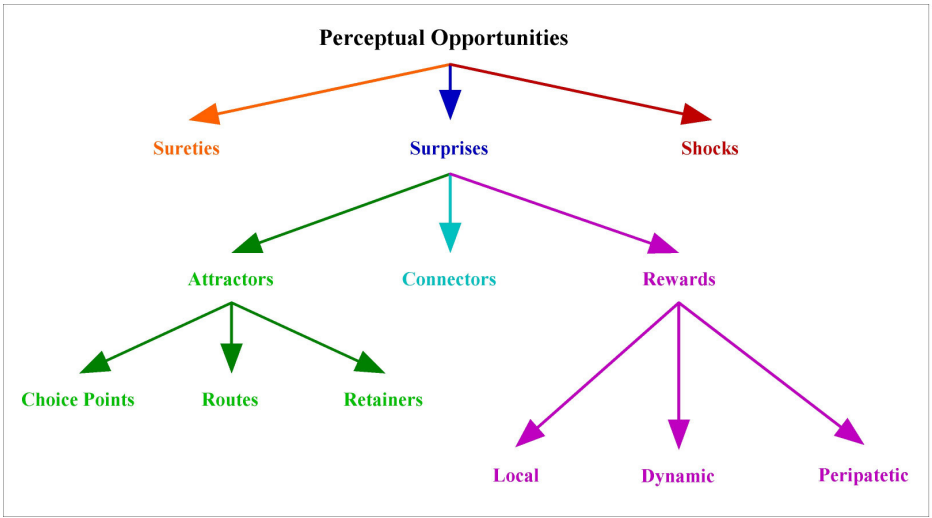


Figure 4.4: Classification of Fencott's Perceptual Opportunities for Virtual Environment Design. The player notices Sureties, Surprises and Shocks and is guided through the VE through different kinds of Surprises.

These perceptual opportunities form patterns which guide the player through the VE, embedding interaction opportunities and narrative in its content (Fencott 2001a, 2001b). In the same way, we can guide novice designers towards understanding how to create interesting and narratively rich interactions by providing inspiring content. The content of the 3D world and seemingly independent interactions of other objects provide Fencott's *sureties* and inspire our users.

4.4. Synthesis

The guidelines that we distilled from this research must be related to our constructivist values and incorporated into our tool design. The fact that experts often create external representations and revisit these throughout the design process ties in with values of multiplicity and reflection. Experts use their sketches as a reference for thinking and reasoning about their designs. Research on external representations in general suggests that they are useful in promoting reflection and a deeper understanding of their subject.

All expert designers surveyed called for early prototyping and the ability to communicate designs. A VR interaction authoring tool that provides the ability to prototype early in 3D, alongside the simple external representations which can focus on specific aspects of the design, accomplishes this. The prototype and representations can be used to effectively communicate design ideas to other members of the design team. In addition, exploration and control are fostered by this, as designers achieve immediate feedback about their

ideas in 3D, and can interact with the 3D Window and the juxtaposed alternative representations to play with their designs (without having to pass the work onto a programmer to see it realized). If 3D content is not yet available, limited prototyping can be done with the simpler representations.

In Fencott's (2001a, 2001b) terms, we can support end-users technically by providing the sureties of a world that they and the player can engage in and reducing the shocks which novice designers might inadvertently include in their designed worlds. We can also support end-users in creating effective designs by providing a space and cognitive support for them to design the surprises which lead the player around the world. This support should include not only technical advice and mechanisms for creating working interactions, but also higher-level support for creating more expert designs.

Most of the guidelines for VR authoring are similar to 2D interface guidelines, with more focus on navigation and orientation, and designing for the first-person perspective of the player. Interactions can be decomposed into basic building blocks, which the designer can then connect flexibly into complex interactions. This dovetails with the constructivist idea of using simple atomic parts. A design tool which makes it easy to create simple interactions within a VE helps to accomplish this; and it encourages exploration, as playing with new interactions is easy and presents immediate feedback. However, support must also be provided for composing the elements of a design into an effective, more complex whole, both in terms of larger system design and in terms of individual interactions. Research on effective usage of multiplicity addresses this and suggests linking of representations both visually and cognitively to support awareness of conceptual wholeness.

Authoring systems directed at novices tend to address technical difficulties and bridge novice understanding of terminology and programming concepts very well. However, they do not support higher-level cognitive understanding of expertise in the domain. An ideal authoring tool should provide support at both these levels. In the next chapter we describe our own study with novices in the area of VR design, where we observed their specific problems and ways of working in this domain.

5. Classroom Observation of Novice VR Authors

In this chapter we describe our first study, an exploratory investigation into the practices of novices in VE authoring. In Chapter 4 we examined expert practices and advice for supporting novices, with the intention of applying these to our tool and design aids. However, we needed a more specific understanding of the needs of novices authoring interactions in VR. Therefore, we observed novices designing VEs and examined the artefacts that they produced.

We used participant observation techniques to conduct this study because they provided us with the most appropriate measures for examining novices at work. They are suited to exploratory research, as the practices of subjects are examined without experimental manipulation and with minimal interference from the observer. In this chapter we first provide a brief introduction to our method. Then we describe the broader research context of the study. Finally, we discuss the study and its results.

5.1. Methodology

Participant observation may be defined as qualitative work which aims to describe the practices of a community in a detailed manner, while participating to some extent in its activities (Spradley 1980). Observers typically immerse themselves in a community to study its practices, rather than create a formal research setting. Various complementary techniques, such as observation, interviews and examination of artefacts, tend to be used.

Our study involved observation in a classroom setting. The community were undergraduate students participating in a course that was partially designed and conducted by our research group. As part of this, we provided them with instruction in VE design. Therefore, we managed the environment and the practices that we were observing. This control was necessary to achieve our purposes. We had to observe novices at work to understand their challenges in VR authoring. Therefore, by definition they would not have performed VR design before, and required instruction to begin learning about it. Once subjects had been provided with information about VE design, and guidance in best practices, we could observe how they used this information. We could also observe conceptual and practical problems which impeded their learning.

The techniques that we used were observation and analysis of artefacts created by the subjects. We observed students designing a VE, while providing some guidance during this process. We also analysed the design documents which students created, and the design aids provided with the documents, to understand the extent of their success in creating an effective VE.

5.2. The Research Context

This study took place in the context of two larger undertakings, which provided constraints on our research. Firstly, it was part of a research project, the Collaborative African Virtual Environment System (CAVES) project, which constrained the technologies that we could use and had its own research aims. Secondly, the students that we observed performed VR design as part of an Interactive Multimedia course. Therefore, they had course requirements which were not part of our research.

5.2.1. The CAVES project

The CAVES Project was a joint venture between the Collaborative Visual Computing (CVC) Laboratory at the University of Cape Town and several commercial and research partners. Its goals were to investigate how to make effective collaborative virtual environments (CVEs) and develop a methodology for this; to develop low-cost tools for supporting CVE authoring by non-programmers; and to develop VEs and content using these tools that would be relevant to a South African audience. Part of making the project relevant to South Africa's needs was to investigate low-cost options, since the large costs of VR is one of the things that make it inaccessible to many audiences. Because of this constraint, all of the VEs created under the aegis of this project were primarily desktop based. The project ran for three years and delivered an authoring tool, VRDirect. Part of the purpose of the broader study conducted with the Interactive Multimedia course was to test this tool and its interfaces (Beirowski et al. 2004). Therefore, the VEs designed by our subjects had to be implemented using VRDirect.

5.2.2. The Interactive Multimedia course

We observed students in an Introduction to Interactive Multimedia course provided jointly by the Multimedia Education Group and the Computer Science Department at the University of Cape Town. The course was offered to Film and Media Studies students who worked in groups. It consisted of forty-eight contact hours, including lectures, tutorials and assisted practical sessions.

For their course assessment each group had to accomplish several objectives related to authoring an interactive 3D mini-game. This game would fit into an existing role-playing game, Handover Street, whose style and setting had been designed by the course coordinator, Dr Marion Walton, and which had been partially implemented (textured geometry and some actors had been created). The game was set in a fictional criminal underground world in South Africa. Players could be thugs, con artists or sneaks. Students had to create one mission for the player character. Their objectives included a formal design document, a game web page, designing and implementing the story and interactions for the mini-game and designing and creating content for the game in the form of digital sound and images.

The genre of a 3D role-playing mini-game is one of the more complex VR applications and has more stringent requirements for plot, character development and look-and-feel than other VR genres (Crawford 1982). We were only interested in how students authored interactions, which design aids they found most useful, and what problems they encountered. Our reflections below only cover these aspects. The time that we had with students (and the time that they had to focus on interaction authoring) was curtailed by the demands for the rest of the course.

5.3. Subjects and Procedure

Ten undergraduate Humanities students took part in the course, none of whom had programming or VR experience. They were divided into three groups, with a CAVES researcher attached to each group. We observed the students during three one-and-a-half hour classroom sessions; the first was a programming tutorial and the others were group design sessions. The role of the CAVES researcher was to provide guidance about the use of design aids, and to make the students aware of VRDirect constraints. Students were provided with information on various design aids and were given an introduction to programming concepts and how to program basic interactions.

During the programming introduction class and the design sessions, we conducted participant observation. We participated in the activities of the students by assisting them with their programming task during the first session, and by informing them of design possibilities and constraints in the next two sessions. We wanted to interfere as little as possible. Therefore, we limited our assistance to information about the capabilities of VRDirect, and highlighting information on the design techniques provided by the course coordinator. Although this study was exploratory and therefore was not based on hypotheses, we created observation guidelines to highlight the kind of information that we wanted. We recorded our observations in field notes, using the guidelines to focus our efforts. These are described in Table 5.1. Due to the nature of the course, we were unable to observe the students' entire design process. However, we were able to gain many insights into the practices and problems of novices during the sessions that we attended. These are discussed in Section 5.4.

The major deliverable for this section of the course was a design document detailing relevant information about the mini-game. Design documents are formalisms used by game design teams to communicate design ideas and game mechanics to all members of the team. They are often used as specifications for a game and include reference material, such as floorplans, character sketches, flowcharts and storyboards. Design documents are lengthy paper documents created in a modular fashion, which typically have sections on game setting, look and feel, style and genre, characters, narrative, interactions, music and dialogue, and marketing. They are useful when a large team creates a game over a few years, as they provide a common reference to

which all team members can refer and contribute (Saltzman 2000). Students were instructed in the use of design documents. After they were handed in, we analysed those parts of the design documents which referred to interactions in the 3D games. This is discussed in Section 5.5. As with the participant observation, we created a set of guidelines for analysing the design documents, which are also displayed in Table 5.1.

Guideline	**Details**
Participant Observation Guidelines	
Student attitudes towards and success employing programming concepts	Feelings about the novice programming system used to teach concepts, feelings about the example programming task they worked on and successful completion of task
Student design process and discussions about game interactions	Feelings about process, focus areas, awareness of non-linear narrative and player freedom of movement
Student interactions with design aids, e.g. floorplans, flowcharts	How design aids were used and which were most effective
Design Document Analysis Guidelines	
Design aids included in document	Which design aids were included, the accuracy and usefulness of the aid
Game interaction specification	Completeness and coherence, consideration of non-linear narrative, consideration of player freedom of movement

Table 5.1: Guidelines for conducting Classroom Observation and Artefact Analysis.

5.4. Participant Observations

We structure our descriptions of the participant observations according to the guidelines in Table 5.1.

5.4.1. Student attitudes towards and success in employing programming concepts

Students were introduced to basic programming concepts with *Alice 2* tool (see Section 4.3), which uses 3D worlds to teach programming concepts. Students were then asked to perform a simple programming task using *Alice 2*. The highlighted concepts were variables and logic structures (e.g., if-then statements and loops). While most students seemed to understand the programming concepts, they had considerable difficulty applying them.

Alice 2 provides natural language and a drag-and-drop interface for program specification. Control statements can be dragged onto the working space and their details added. However, even with this help, students were unable to translate the problem description into a workable programming specification in the time provided. They expressed frustration about the program and their ability to work with it. We had

expected the programming example (object rotation and movement) to be easy for most students; however, we underestimated the extent to which they would be forced to learn a new way of thinking. Unfortunately, because of the course requirements, no more programming instruction could be given to the students. In addition, at that time, VRDirect only had a Python-based scripting interface. We had planned for students to implement their mini-games in VRDirect with some assistance. It became clear that this would not be possible. Therefore, we decided that a CAVES researcher (Charlene Elliott, formerly Beirowski) would program one of the mini-games, using the students' design document as a specification.

During the following sessions, researchers introduced the students to several design aids which would assist them in specifying their game interactions in a detailed enough way for the programmer. These included the use of simple pseudo-code and thinking of interactions in terms of events and triggers.

5.4.2. Student design process and discussions about game interactions

Students seemed excited by the idea of creating their own 3D game, and were willing to engage with the provided techniques. They brainstormed a lot about their mini-game, generating many creative ideas. The challenge was not coming up with ideas, but harnessing that creativity and understanding the restrictions that time and technology would place on implementation. In general, students were inspired by the story and content aspects of creating a 3D game, but had difficulty considering the low-level game mechanics and specifying the basic interactions that would result from their ideas.

During the first session, students spent much time considering various scenario options and produced multiple narrative possibilities. The desire to accommodate everyone's ideas led to a tendency to engage in detailed discussion of minor interface issues, rather than focussing on higher-level goals of the game. For example, one group spent much time discussing how to show that the player character had received a phone message. The students did spend some time discussing game play, especially how the mood and setting would impact on the player character. This led them to consider some higher-level interactions. For example, students considered how to link branching narratives to player choices, e.g. how winning or losing a fight would impact the player's options later in the game, or the penalties for taking too long at a task.

At the beginning of the second design session, the course coordinator reminded the groups to focus on higher-level goals. After this they focussed more on the game interactions. During initial discussions, students realised that they would have to refine their missions before being able to plan detailed interactions. By the end of the session, all students had finalised their chosen missions and begun discussing interactions. They considered player movement through the game and the broad interactions that this would entail.

We were interested in two particular aspects of interaction design: consideration of non-linear narrative options and player freedom of movement. Both are important characteristics of VR that distinguish it from film (with which Film and Media students have much more familiarity). Non-linear narrative stems from providing different gameplay options to the player, e.g. entering a building from two different entrances or playing different kinds of character. Freedom of player movement is a similar concept; it is distinguished from non-linear narrative because it does not relate to gameplay options but is about the player's ability to explore the game in a way that is not related to the designer's narrative goals, e.g., the player deciding to walk in the opposite direction to the action. We were interested in whether students showed an awareness of these aspects of interaction design, and how they were handled.

Non-linear narrative was considered quite frequently. Students discussed various player options and their effects. This was probably because branching narratives are an important part of role-playing gameplay and had been highlighted during lectures. In addition, students were required to consider multiple player characters. Conversely, freedom of player movement was not well considered. Students tended to talk about the interactions in terms of various paths through the game related to narrative options, but did not consider what would happen if the player did not conform to these options. They focussed on the stories that they would be telling with their games and seemed unable to consider alternative forms of interaction with their worlds.

5.4.3. Student interactions with design aids

Students were taught about various useful artefacts for designing VR and games. These were floorplans or maps (i.e., diagrams showing relationships between spaces and elements), flowcharts (i.e., network diagrams showing flow of data or action), storyboards (i.e., sequences of annotated images showing changes in VE scenes) and use case scenarios (i.e., descriptions of typical user actions with a system). They were encouraged to think spatially about their games, so that they would fully use the 3D technology.

The floorplan worked well as a base for understanding interactions. When the groups started discussing their ideas, they all used a floorplan as a shared focus around which to centre the discussion. In one group, a student drew a rough floorplan of the 3D world and used it to explain her ideas, with the others contributing in the discussion. At some point another group member drew a textual storyboard to represent the ideas. A third member drew a flowchart while the others were talking, describing the story and showing narrative branching. In this group, each member created a different diagram to illustrate their discussion, but the floorplan was the only artefact to which they all referred. When they began discussing gameplay, they examined each location on the floorplan and discussed what would happen there. They used arrows to indicate player movement and drew icons for other characters or objects to indicate interactions.

In another group, one member was chosen as the scribe. He wrote notes as the discussion progressed, using arrows to connect related ideas. He illustrated some of the ideas on a floorplan which he drew, with arrows indicating interactions. The other members began pointing at the floorplan and using it in further discussion. They expressed that they found the shared representation very useful, as members could draw attention to specific details and alter them in real time while communicating. The floorplan was especially useful as it provided a visual reference to anchor the discussion. However, some parts of the interactions seemed more difficult to indicate on it as there was no spatial element to anchor them, such as timed events.

Students grasped the concept of flowcharts easily and some created story diagrams that were similar to the flow diagram formalism before this was introduced. For instance, some of the notes written by the group with the scribe were very similar to the flow diagram formalism. The flowcharts created by students during discussions tended to be flowchart/statechart mixtures. Unlike typical flowcharts, they did not show decision nodes or indicate the process of the game; they instead showed states of the player (like having an access code) and the transitions were links to the next state of the player, sometimes labelled with the action needed to accomplish that state. They therefore showed the structure of the high level game interactions.

Storyboards were not drawn much during our observations. When they were, they tended to depict events or actions associated with particular objects or spaces, rather than longer narrative sequences. Use case scenarios were initially discussed and used to consider how the player would interact with the game; however, students stopped using them because of the time required for each one.

5.5. Analysis of Artefacts

Because the students were working on a course, they kept all artefacts created during the design sessions. Therefore, we could only analyse the artefacts which they handed in with their design documents. Although the students worked with most of the design aids to which they had been introduced during the design sessions, they did not hand them all in. For example, while all groups drew floorplans during their discussions, only one group handed them in; these floorplans had been cleaned up and did not include annotations made while designing (indicating spatial triggers, movement direction, etc). In addition, although groups drew storyboards and considered use cases, none were handed in. These design aids, in any event, had not been very useful to students and had not added much to the design; it therefore seemed appropriate that they were not submitted. However, the lack of floorplans surprised us and impeded our analysis of the design aids, as they had been very useful during the design sessions. As stated earlier (Section 5.4.3), the floorplans were a central external representation during the design sessions, as all students annotated them and used them to guide discussion of the interactions. Therefore, they provided constraints for the design

discussion and aided communication between group members. During our analysis of artefacts, there was often confusion about spatial details, which might not have occurred if the floorplans had been included.

We have included the parts of the design documents which related to interaction specification in Appendix A. We only analysed documents for two of the groups, as the third group's submission did not provide sufficient interaction details to be included in our analysis. The two groups which we examined will be referred to as Group A and Group B. Before examining the design documents, we decided to focus on design aids and textual interaction specifications separately (as indicated in Table 5.1). However, the two design documents specified interactions in such different ways – one used purely textual descriptions and provided a separate flow chart and the other created a flowchart in which they embedded the textual interaction descriptions – that we decided to examine each one as a whole. The interaction specification provided by Group B was more detailed and specific than that of the other groups, and their scenario was more easily programmed using VRDirect; therefore, their game was chosen as the one that would be fully implemented.

5.5.1. Group A

The artefacts for Group A can be found in Appendix A1. Discussions of artefacts reference their page number in the appendix. Group A provided a game treatment; a series of flowcharts; and related floorplans to describe interactions. In their game, the player is a thief who must steal a car and deliver it to the Under Lords. Group A did not consider other player characters (i.e., the con and the thug). In order to steal the car, the player must get past a security guard into the Tourist Complex. If the player resists being searched, he will be caught when trying to leave later. He can choose to abort the mission at this point. Once inside, he must find Metro near the bar in the Moroccan Club, give him a password, and get an access card to enter the Security Building. The player must then enter the Security Building through the back door, avoid security guards, and steal a disguise and coat hanger to break into the car. There are keys available, but they do not work. The player must then find the car, break in and drive away.

The treatment provides a high-level walkthrough of the game plot, with some player background and character descriptions (page 1). The flowchart suggests that the mission described in Group A's game can be rejected and the player can continue with the larger Handover Street game. An arrow running down the length of the overall flowchart indicates that the game runs on a timer, as the mission must be completed by 2am (although there is no description of consequences of being late).

The flowcharts and floorplans created by Group A describe the interactions of their game quite clearly, and in more detail than the treatment. They first provide an overview flowchart, which describes the whole game (page 2). Like a sequence diagram, this flowchart contains superstates which are flowcharts themselves. Each bolded block in the chart refers to a sub-flowchart which describes a scene of the game. Where the

scene corresponds to a physical location, the designers provide a floorplan of the location. These floorplans are simple maps, sparsely annotated with textual descriptions of interactions that will happen. The usage of superstates in the overall flowchart indicates some sophistication and awareness of the problems of displaying complex information. The connection of the flowcharts with the floorplans shows that students could conceptually link different design aids. The timer arrow indicates awareness of a third dimension, time, which is not easily conveyed with any of the design aids, and an attempt to show this. Therefore, students understood the usefulness and limitations of the design aids and were skilful enough to try and customise them. They used the flowcharts correctly to indicate multiple interaction options at various points in a way that would be much less apparent in a purely textual form. However, the indications of linearity in the treatment were carried over to the flowcharts.

There were mistakes in the usage of the design aids, usually by leaving out details which would be important for implementation. These were apparent, not only in the graphical representations but also in the textual descriptions. Because the students did not have any direct experience with implementing their designs, they were much less likely to be aware of these holes. For example, although a timer is shown on the overview flowchart (page 2), there is no description anywhere else of what this means in terms of the game mechanics. There is no indication of the time at which the game starts, and therefore no idea of how long the player has to complete the game objectives. The game timer is only mentioned once elsewhere: on the Car Theft map (page 10), there is a section on sound, which states:

The tempo picks up as the player starts to run out of time on the final hi-jacking task.

However, for this to be implemented there would have to be some idea of how the time is kept and the starting time. Therefore, Group A was aware of the time issues, but unable to address them completely and unambiguously. This could be because none of the design aids addressed time. Another example is when the player approaches Metro in the club (page 6). If the player does not give the password correctly, Metro rejects him. There is no indication of whether the mission fails at this point or the player can try again.

There was also some inconsistency in the flowchart syntax. For example, in the overview flowchart, the blocks that are not bolded can either refer to player options, to interaction points, or to game actions based on player choices. In all of the flowcharts, different options are included in the boxes depending on what is needed to describe further interactions, such as dialogue, player options or player action descriptions. Action descriptions labelled a, b, etc, can signify player options or steps to accomplish one task. This can be seen in Figure 5.1, which displays an extract from the Car Theft flowchart (page 9). Options 2a and 2c are steps that the player must complete, whereas Option 2b is an alternative to 2a. The links in the flowcharts are also confused at times. Students leave out connections between boxes, or have two links which lead to the same

conclusion but different boxes. This can also be seen in Figure 5.1, where the box containing 2d captures the same step as that described in 2c.

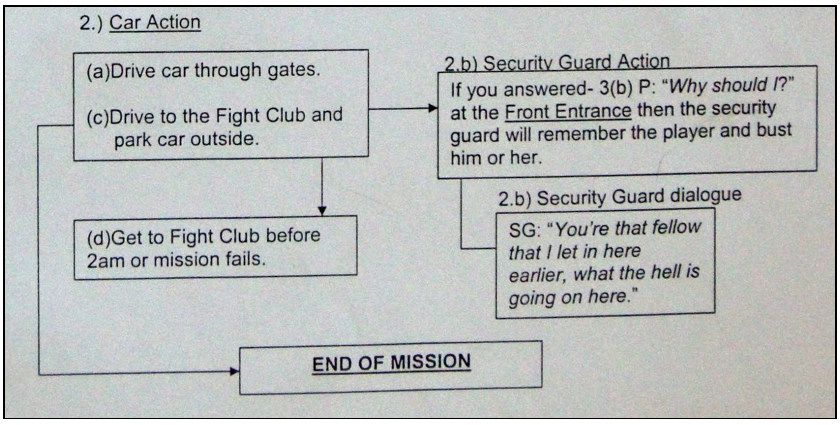


Figure 5.1: Bottom section of Car Theft flowchart provided by Group A, showing some inconsistencies.

These problems have three likely causes. First, the students were not skilled at creating the design aids themselves. In most cases, this was the first time they had created flowcharts and maps, especially ones that needed to indicate complex, interrelated dynamic interactions and their effects. Second, the design aids were not linked automatically, which would make inconsistencies more obvious. Third, because the students did not have to implement their designs and were working according to a course schedule, they did not have to take the time to make sure that their aids were consistent. They were also not made aware of the consequences of sparse or inconsistent details.

The same problems can be seen to some extent in the usage of floorplans. While the floorplans of each location are clear, detailed and connected well to the flowcharts, there is no indication as to how they fit together, and there is no overview floorplan to indicate the relative positions of each location. In addition, the amount of detail on the floorplans is inconsistent. Some describe ambient details such as sound and some have arrows indicating player movement through the area (e.g., the Car Theft floorplan on page 10), but in other cases these details are absent (e.g., the Security Building floorplan on page 8). There is also no floorplan for the Moroccan Club scene, which may be because the students had timing issues with their hand-in deadlines. Figure 5.2 shows the floorplan provided for the Security Building. It indicates important areas in the building and the player starting point. However, it does not show the positions of security guards, their possible movement paths, or the placement of objects in the Change Rooms.

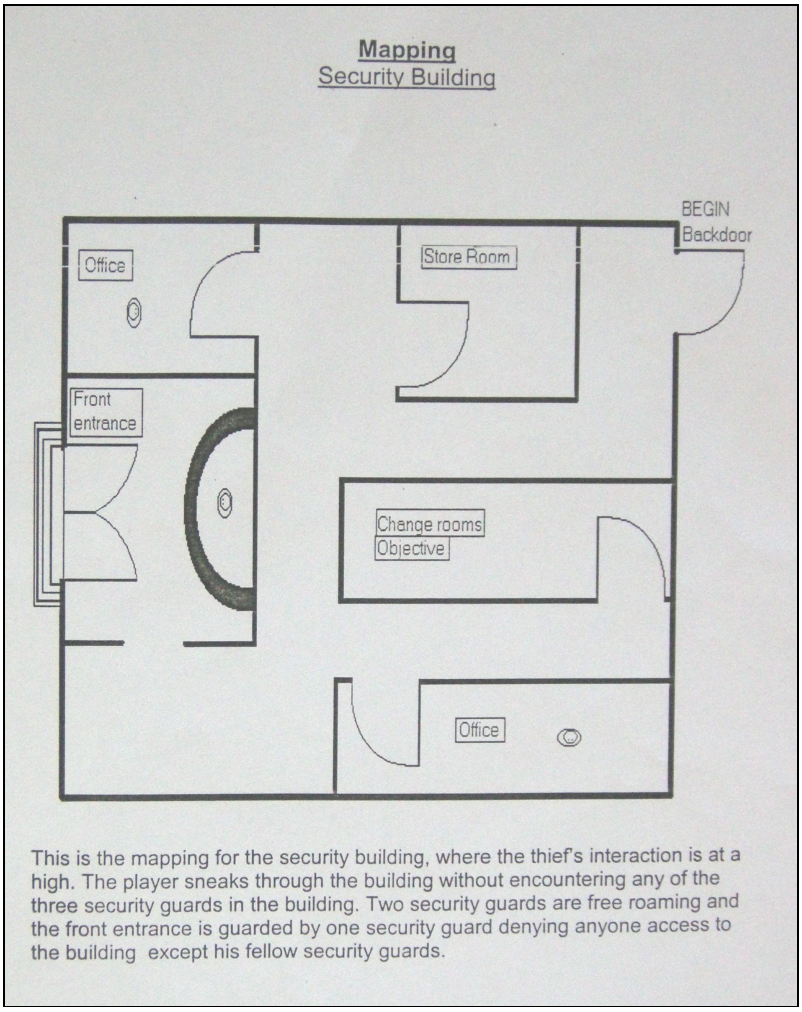


Figure 5.2: Annotated map provided by Group A for the security building.

Although the dialogue options and their consequences are well described, the flowcharts include few other low level descriptions. In providing a good impression of the look and feel of a game, these details are unimportant. They only become obviously necessary when the game is implemented. For example, the security guard stops the player in the first scene and searches him (page 4). While it is easy to imagine this scene, the description is not precise e.g., there is no indication of how close the player must be to the security guard to start the interaction. The floorplan (page 5) shows the layout, but provides no scale. This kind of specific detail is easily overlooked as it seems obvious, but it is important for implementation.

Another example is the series of interactions in the Security Building (flowchart page 7 and floorplan Figure 5.2). The player must sneak past two roaming security guards in the building. However, there is no indication of the security guards routes or the conditions under which the player will be seen. The security guards are also not included in the character descriptions (page 1). The lack of dynamic floorplans, which show

movements of objects, contributes to the difficulty of placing dynamic objects. When the player is in the Security Building, the floorplan shows the room with the objectives of the scene: a locker containing a disguise, car keys and a coat hanger. However, there is no description of the room, how the locker is organised, or how the player grabs the items.

In the Car Theft scene (flowchart page 9 and floorplan page 10), the students state that if the player is seen breaking into the car, he will be busted. However, they do not provide details of a character able to see the player breaking into the car, what happens if the player is busted, or how the player activates the coat hanger. This connects with designers only considering a winning walkthrough by the player. Figure 5.3 displays part of the Car Theft flowchart, showing the lack of details.

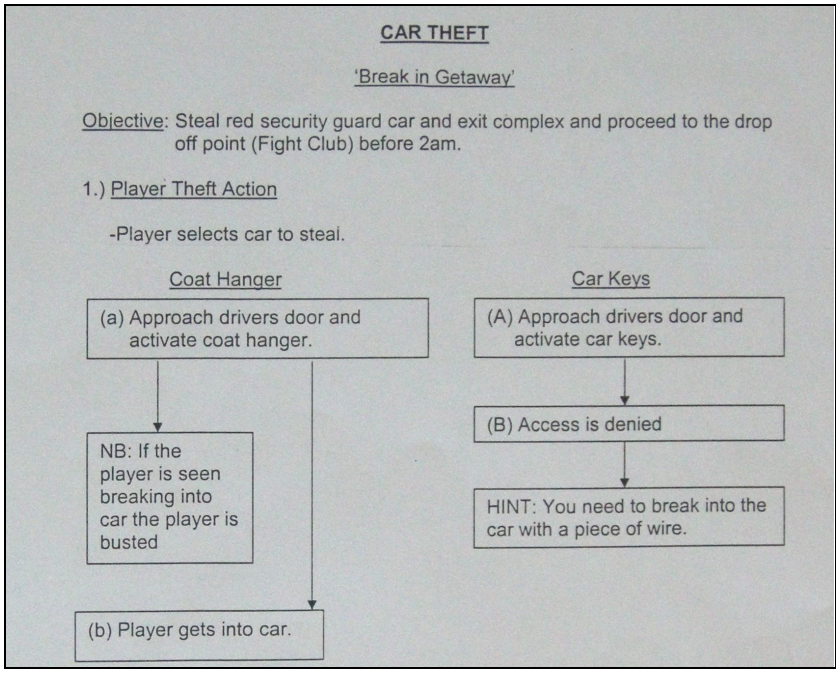


Figure 5.3: Top section of Car Theft flowchart from Group A.

Overall, Group A created a straight walkthrough, with few possibilities of deviating from the action. Alternative dialogue options are well described, but these do not have consequences except to fail on the mission. The students did use the flowcharts effectively to indicate the typical flow of interaction, its look and feel and its connection to the game narrative. Their floorplans were helpful in understanding the spaces in which interactions would happen. However, they left a large amount of detail out of their descriptions, making it difficult for the 3D game to be implemented accurately. Game designs must be precise, both in terms of exact interaction details and in terms of the appearance of objects and locations. A programmer would have to make many creative decisions on the designers' behalf because of what was not specified, which might change the game in ways not intended by the designers.

5.5.2. Group B

The artefacts for Group B can be found in Appendix A2. Group B provided a much more typical formal design document than that provided by Group A. The interactions of their game were described textually in various sections of the design document, under Gameplay, Player Experience, Narrative Plot and Interactions, Interaction, Characters and Music. They also included a flowchart. They did not include any floorplans of the game. The only place where the spaces of the game were described graphically was in the Reference section. As well as providing reference sketches of some of the characters of their game, Group B provided sketches of various locations. However, these were not clear enough to be used as floorplans. Figure 5.4 shows a sketch of the bar area of Club Dune, where much of the action takes place. While it provides an idea of the look and feel of the area, it does not include information about the size of the space, location of objects and characters, or possible movement paths.

Figure 5.4: Sketch provided by Group B showing bar area of Club Dune.

In Group B's game, the player can choose to be a thief, a con or a thug, which results in different interactions. The objective of the game is to steal a gun from a safe in Club Dune for Mr V. The player must get into the club by paying the Bouncer, trickery or breaking in through the staff entrance. Once inside, he must persuade the Barman to point out Natasha, who is the contact. She directs the player to steal a disguise and tells him that the gun is in a safe upstairs (guarded by a bouncer called Stop Sign), giving him the code. The player then goes upstairs (the thief climbs a vent and the other two characters use trickery), opens the safe and retrieves the gun. Stop Sign arrives and starts shooting. The player must engage in gun battles with him, the club owner and a gangster named Khan to get out of the Club.

Group B included much more interaction, narrative description and overall detail than Group A. Their design document contains multiple overviews of the game action. The section on *Gameplay* (page 11) describes the player characters and the consequences of choosing one. *Player Experience* (page 11) provides a high-level walkthrough of the game plot. *Narrative, Plot and Interactions* (pages 11-12) provides a rich narrative

background to the game. Interactions are described in detail in the *Interaction* section (pages 12-17), and labelled with the location in which they occur. Like Group A, the students focus on the dialogue of the game and many of the interaction options are centred on dialogue. The section titled *Characters* (pages 17-18) describes the characters in the game, their motivations and major interactions. *Music* (page 18) describes the music used in the game, its narrative importance, and indicates what the player can hear in each location.

As mentioned earlier, Group B's interaction descriptions were much more accurate and detailed than those of Group A. For instance, they indicate the triggers for various actions, e.g.

When the player moves within 1m of the women "Press Button" will appear on screen (page 12)

Group B provided descriptions of interaction branching on multiple levels. They included different interaction possibilities for the three player characters (e.g., if the player breaks into the club through the back door, he must knock out a security guard; each character knocks the guard out for a different period of time), for player action options (e.g., whether the player enters the club using the front door or the staff door), and for player dialogue options. They often carried these through various scenes of the game. For example, if the player enters through the staff door, the bouncer will look for him during the Club scene.

However, although they obviously understood the need for details, they also left information out. For example, when the player visits Natasha in the store room for the second time (page 15), there is no indication of how close the player must be to the door to trigger the cut scene. During the gun battle, health can be gained from alcohol bottles dotted around the club area (page 17). However, the number of bottles is not specified, nor is the extent to which the player's health is improved.

Neither group provides descriptions of what characters are doing when the player is not interacting with them, i.e. their default resting actions. There are also no descriptions of any characters in the bar area of Club Dune apart from the barman and Natasha. Yet, this is supposed to be a busy club; later there is a description of people running out of the club when the fire-fight starts. Decisions about these characters, which would add to the ambiance of the game, have to be made by the programmers.

For spatial specifications, there are no descriptions of the paths taken by various characters when they are patrolling areas. For example, when the Bouncer comes looking for the player, the only description is:

The bouncer from the Staff entrance will know (sic) *be alert and will be patrolling around the club looking for you.* (page 15)

In some cases, the students prescribe actions for the player, which violates the core principle of player freedom, e.g., *Khan will move around the entrance and dance floor, while the player will be near the stairs and the rear of the club.* (page 17) However, unless the movements of the player are circumscribed, the player will be able to engage in the gun battle anywhere in the room. The interaction description includes many references to cinematic shots. For example,

CUT to CLOSE UP shot of player's hand slyly giving the bouncer the money. Then Medium Shot of player walking into the club. (page 13)

The students use these to control the narrative of the game. However, while creating mood, they also remove player control. The player is unable to undertake the story in her own way. In addition, camera control may be taken away from the player at inconvenient times.

The flowchart that was provided is much less accurate and detailed than that provided by Group A. An excerpt is provided in Figure 5.5. In some cases, the flowchart conflicts in details with the textual interaction description. For example, the flowchart indicates that the thief character will climb back down the vent after opening the safe and meet StopSign in the store room. However, the text indicates that all three character types meet StopSign near the safe. This is shown in Figure 5.5, blocks 29 and 30. The text states the following for all characters:

Player will then walk out of the office and into the lounge. When the player is 2m from the office Stop sign will come into the office carrying a 9.mm pistol and he will begin firing at you. (page 17)

In other cases, the flowchart leaves out interaction options. For example, there is no option for the Bouncer catching the player in the Club (page 15 in the text). In general, the flowchart provides a linear walkthrough of the narrative flow of the game, and does not consider what happens when the player does not follow the narrative. In some cases, the flowchart shows the different paths of the three player characters, and at other times it does not. For example, in Figure 5.5, block 31 is all that describes the gun fight. In the text, the description is as follows:

Khan will open fire with his A-k and spray bullets everywhere. The player will die and have to restart in the head office at the safe if he is shot more than six times as the power bar will indicate…Unlike the club owner it will take more shots to kill Khan. Thug = 5 Con = 6 Sneak =7. (page 17)

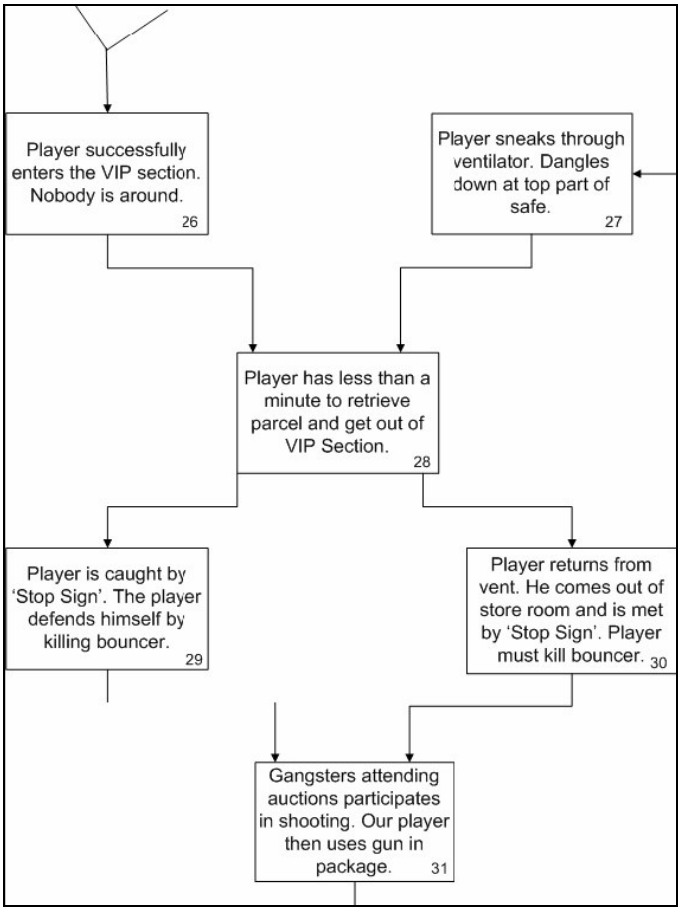


Figure 5.5: Section of the flowchart provided by Group B, showing inconsistencies with the textual descriptions.

There are several details in the text which could be shown on the flowchart, but are not. For example, each player character needs a different number of shots to kill Khan, and if the player dies he will restart at the safe. Because of all of these inconsistencies, it would have been better for Group B not to include a flowchart. Unlike with Group A, the flowchart does not add anything to the interaction specification. It is linear and does not indicate any results of the player not following the 'winning walkthrough'.

The game of Group B was chosen for full implementation over Group A. Therefore, the programmer who implemented the game in VRDirect could make comments about this process. She found that it was time consuming and difficult to use the textual descriptions of interactions, especially for finding required props and actors, their appearance and their exact locations. The interactions of Group B, while less linear and more detailed than those of Group A, were harder to understand and debug because of their textual format. They were described in various places in the design document, not only in the *Interaction* section. The lack of a floorplan made positioning all the actors and interactions much more difficult. The programmer had to transcribe the interactions into tabular form so that she could associate all conditions and events with their

actions. As mentioned above, many interactions were not specified in sufficient detail, so she had to make creative choices about what the designers of the game intended. She did find that options around dialogue were well described, and that the students naturally created if-then statements for dialogue options.

5.6. Novice Strengths and Weaknesses in Interaction Authoring

In both Group A and B, the games produced were interesting and expressed with attention to creative detail. Both groups described their content in a way that enriched the narrative of the games. This makes it especially unfortunate that they were unable to create a complete specification. By not expressing all necessary details, they were unable to control the implementation of their creativity and the programmer needed to assume details.

Comparing the mechanisms for specifying interactions, the flowchart used by Group A added ease of understanding, but lessened the detail provided. The textual descriptions used by Group B were difficult to parse, but included more detail and alternatives. Both groups wrote their interaction descriptions more like a player manual than an implementation specification. Although both groups used annotated floorplans successfully to describe interactions during their design sessions, neither provided these completely in the design document. This suggests that neither group considered the floorplans useful enough to include, although it could also be that they take longer to prepare in a more polished format and students did not have the time. In the case of Group B, the programmer found the lack of floorplan to be a significant disadvantage in understanding the game space. From the observations and examination of artefacts, we identified students' strengths and weaknesses in their game design process. These are listed below.

Strengths observed:

- Willingness to experiment with new methods and techniques
- Successful use of the floorplan as a visualisation during the design process
- Usage of if-then statements to describe interactions
- Understanding of the flowchart formalism and creation of relatively accurate narrative flow

Weaknesses observed:

- Forgetting to state all details and behaviour explicitly
- Linear designs, where interactions are described as a walkthrough of the plot
- Inability to consider alternative paths that a player might follow
- Inconsistent, inaccurate and linear flowcharts
- Inability to document interactions clearly and consistently
- Difficulty working with programming constructs
- Forgetting to list and describe all assets

Students' main problem was trouble conceptualising interactions that diverted from their planned narrative. They did not consider what might happen when the player did not conform to their design, and often left out alternative paths. This occurred even though they had been taught about interactions and how to specify them. Using a design document tended to amplify this problem, as designers created a story of the VE by describing its look and feel. They became tied to their stories and unable to visualise alternatives. The headings provided by the design document did not assist students in organising their interactions in a way that would be easily described programmatically. The design document also did not assist students in understanding the mechanics of authoring interactions; they therefore specified interactions that were too complicated for VRDirect to handle.

Our results are very similar to those of Pane et al. (2001), who conducted a study where children were asked to describe the operation of a Pacman game. The authors found that participants used if-then statements most frequently to describe interactions; two-thirds of them used diagrams for early planning and layout; they often did not use programming constructs like loops successfully; and they often left out important details or were imprecise in their descriptions. These similarities provide evidence of the generalisability of our study results to other novices and other programming design tasks.

5.7. Synthesis

Table 5.2 provides a list of design issues that we observed in novices and their artefacts. We also describe possible support for addressing these issues, based on the research examined in Chapters 3 and 4.

Design Issues	**Details**	**Possible Support**
Non-linear narrative	Considering multiple narrative paths and multiple options for interaction	Reflective scaffolding; multiple representations
Player point of view	Understanding player independence and unexpected behaviour	Early prototyping; the ability to explore the artefact as the player
Programming	Understanding programming concepts	Supportive and intrinsic scaffolding; simple interface with familiar and understandable mappings
Interaction details	Specifying all of the details required to completely describe an interaction	Supportive and reflective scaffolding; early prototyping
Asset specification	Specifying all assets in the detail and form requirement	Supportive and reflective scaffolding; early prototyping
Time and space	Considering and specifying timed and spatial aspects of interactions (e.g., time	Multiple dynamic synchronised representations (time based and spatial

Design Issues	Details	Possible Support
	limits, spatial triggers, movement paths)	representations)
Creation of sureties	Making the world believable, e.g., by adding default actions to objects	Supportive and reflective scaffolding; early prototyping

Table 5.2: Table showing the most important support needs of novices in VE interaction design.

Once again, the usefulness of constructivist values is highlighted. Designers in general benefit from viewing multiple perspectives on a design (see Section 4.1.2). Multiple representations can be used to indicate the extent to which interactions are linear and unconnected. Such dynamic, synchronised representations can assist novices in specifying designs completely by reflecting on them. However, novices may have problems with accuracy and detailing if required to create the representations themselves (as the flowchart experiences indicate). Therefore, help in the creation and understanding of representations is necessary.

If novices can implement simple interactions, these will be understood easily (fostering feelings of control). In addition, with simple interactions first steps in design implementation are easy to take. Immediate feedback about their consequences can be provided through early prototyping and mistakes can be fixed. Using natural language in a sentence structure to describe interactions is well understood by novices (see Section 4.1.2). 3D prototyping with a player perspective (i.e., experiencing the designed game as the player) combined with representations, can address timing errors, and help novice designers to find details that they have forgotten to specify; and combined representations and scaffolding can make novice designers aware of their implicit assumptions. All of these features should help novices to communicate their designs more effectively in teams and enable them to implementation their designs themselves to some extent.

In the next chapter, we describe our initial design of VRBridge and the design aids, based on our research into the practices of experts and novices as described here and in Chapters 3 and 4.

6. Design of VRBridge

This chapter describes the design of VRBridge, our system for authoring VR interactions. Our work until this point has prepared us for the system design. We uncovered the principles to guide the creation of the tool and design aids in Chapter 2, where we developed the Constructivist Design Method. These were atomic simplicity, multiplicity, active exploration, control and reflection. We also described two focus areas which incorporated these principles: multiple representations and scaffolding.

In Chapter 3, we described research into the design aids. The research on external representations provided us with information on which representations work best for different tasks, and how they may be used effectively. The research on scaffolding showed how to support users in learning effectively. In Chapter 4, we examined work on expert practices and supporting novices, especially in design and VR. Finally, in Chapter 5, we described a study to investigate the work of novices in VR authoring.

For the initial design of VRBridge, we focus on the system architecture, the interaction authoring interface and multiple representations. We leave scaffolding and the 3D Window until after the basic system has been evaluated, and describe them in Chapter 8. This allows us to focus on the effectiveness of the representations in describing an interaction design without a 3D Window. If end-users can understand 3D world interactions using only the design representations and authoring interface (i.e., without experiencing them in 3D), this will show us that they have the necessary expressive power and ease of understanding. In addition, by examining end-users working with the system, we will have a better idea of their scaffolding requirements.

In this chapter, we first provide an overview of VRBridge, showing how our constructivist principles become general design values. Then we describe the system architecture and its implementation. We complete the description with the programming model and authoring interface. Thereafter, we describe the design aid of multiple representations. Finally, we show how the parts of VRBridge are linked, both conceptually and practically. Some of this design has been described previously (Winterbottom et al. 2006).

6.1. Overview of VRBridge

The constructivist principles that we have distilled underpin the design values of VRBridge. To foster a sense of personal *control*, we ensure that users can author a VE in their own style. Workflow is designed to be as flexible as possible, with no required order of tasks. This aim is made easier by following the principle of *atomic simplicity* and deconstructing tasks. To promote *exploration*, we provide an interface where interactions can easily be added or removed. These simple components can then be combined to form more complex interactions. Exploration is also encouraged by providing immediate feedback on authoring actions,

which increases end-users sense of personal control (as shown in Table 4.1). Immediate feedback is achieved through multiple representations of different interaction aspects, which do not have to be coupled with a 3D Window. The principle of *multiplicity* is addressed by multiple representations, along with the 3D Window and the authoring interface. The provision of multiple, overlapping views of the design process and domain should help users to easily make the connections between parts of the system and authoring tasks. We also believe that by providing effective design aids, we can help end-users to think in non-linear terms, and *reflect* more deeply and effectively on their designs.

Our research on how designers work (discussed in Section 4.1.2) showed that support for tentative actions is desirable for creative work. A tentative action in interaction authoring can be equated with the ability to create an interaction quickly and easily, and view it in context immediately so that it can be evaluated and adjusted. The same research shows that designers create alternative views or representations of a design. This supports our design aid of multiple representations. Our classroom observation in Chapter 5 and investigations into novice support in Chapters 3 and 4 provide us with specific areas where novices need authoring support:

- Novices forget to express interaction details explicitly. Therefore, the tool should help them to consider all details. Viewing interactions immediately in a 3D world will help to accomplish this, but additional representations and the authoring interface can also assist.
- Novices create linear designs and show lack of awareness of alternative player paths. The representations can make linearity obvious, and allow the user to experience the VE as a player.
- Novices have problems with programming constructs like iteration, although they understand conditional statements when expressed in natural language. They also naturally use an event-based style when programming. As text is better than images for representing procedural information and logical conditions, we will use a natural language, event-based programming style.
- Novices understand the semantics of floorplans and network diagrams, which suggests useful representations to provide. They also have problems specifying temporal and spatial aspects of interactions.
- We can provide the sureties which make the world believable, e.g. textured objects and gravity, in order to provide novices with design inspiration and guiding constraints on creating interactions.
- Novices often have problems in understanding 3D space. Therefore, we use 2D graphical representations to help them in effectively working with VR space.

6.2. Designing the Authoring System

Since our system supports interaction authoring, we are not concerned with other aspects of VR creation, such as content creation. We include a library of objects in VRBridge, which are models with associated

sounds, animations and textures. We also provide 3D world geometry. The player and VE world are special objects within VRBridge. The player is not embodied in our prototype, so it is represented by a camera (controlled by the player), and has no animations. The *World* object potentially consists of multiple models and provides the 3D space. Another point to consider is our focus on low-cost VR which uses ubiquitous technology. Therefore, VRBridge creates a 3D world on a desktop computer, with the keyboard and mouse as input devices. This restriction lessens the immersion of the 3D world, but does not affect the abstract nature of the interactions. VRBridge is designed to be extensible and modular, so alternative input devices can be added and the system ported to a more immersive technology in the future.

6.2.1. System architecture

A core design value for VRBridge was to create a flexible system, which could be used flexibly and extended. Therefore, we designed a modular object architecture, which is displayed graphically in Figure 6.1.

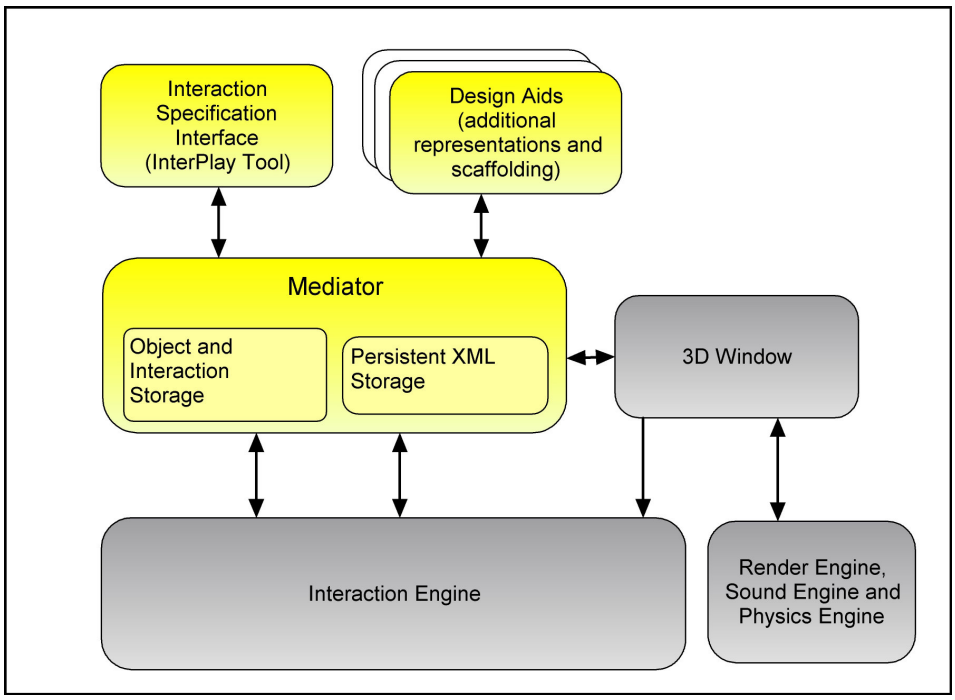


Figure 6.1: VR Bridge architecture, showing the main modules and how they interact. The highlighted modules are described in this chapter, while the greyed out ones are described in Chapter 8. All modules communicate through the Mediator, which stores content and interactions.

It consists of a core module (the Mediator) which handles communication between other modules, and manages and stores the content of the VE and programmed interactions. The authoring interface, the design aids and the 3D Window are separate modules, which are plugged into the Mediator. This allows additional

design aids, alternative render engines and an alternative programming interface to be added easily. The modules communicate through a messaging system contained within the Mediator, which creates a high degree of interconnection on multiple levels. In addition, the Mediator enables loose coupling between modules, as each sends and receives messages to and from the Mediator, rather than the other modules.

Links to all content and any specified interactions are stored in XML files for fast access. When VRBridge is opened, the Mediator extracts required information from the file and loads pointers to the content. Therefore, when the user first views VRBridge, it contains engaging content (e.g., models with animations and sounds) which can immediately be explored. This provides constraints on authoring actions, by allowing the end-user to program interactions immediately, using objects and spaces that inspire design ideas.

Any interactions that are entered using the authoring interface (or InterPlay Tool) are immediately stored in the Mediator and available to the Interaction Engine. Object details (e.g., position) are created or edited using the InterPlay Tool or the design aids. These are also stored in the Mediator. The Interaction Engine is a separate module which communicates mainly through the Mediator; it also receives input from the 3D Window. It contains rules for how objects and the world are changed when interactions occur, and manages the execution process by working out when and which interactions will occur in each frame of an executing VR. It passes information about the effects of interaction on to the Mediator, which makes the required changes to the content and manages the frame loop. In this way, the Interaction Engine is a virtual machine which interprets the binary code produced by the interactions at run time, so that no compilation is required.

Information about the executing VE can either be passed to the 3D Window or to the appropriate design aids. The additional representations can therefore show the possibilities for interaction and their consequences, even if the 3D Window is not running. The 3D Window manages the VE and its interface, and contains generic functions for 3D interaction. These are passed to a separate module, which translates them into specific functions for the render engine, sound engine and physics engine. The 3D Window, Interaction Engine, additional engines and scaffolding design aids are described further in Chapter 8. The InterPlay Tool and additional representations are described in this chapter.

6.2.2. Implementation details and the Mediator

VRBridge was programmed using the object oriented language, c++. Pointers to content are loaded when VRBridge opens. Everything is defined within its own object class, so that it can be easily extended or altered, e.g., each sound is an object with a unique identifier, a name, and a pointer to its file. The Mediator contains a class named Universe, which contains pointers to all of the objects required for the VE in multiply linked lists. It contains lists of VRObjects, Locations, Waypoints and Triggersets, and polymorphic functions to manipulate the lists. The use of multiple levels of linking and connection between objects and lists ensures

that VRBridge is flexible, with multiple access points to structures. VRObjects and Triggersets form the inheritance structures shown graphically in Figure 6.2.

BaseObjects in the VRObject structure contain generic details such as model file name, starting position, and dynamic variables which the Mediator updates as the VE executes. This is accomplished with a virtual function to update each object. Inherited classes contain variables and functions specific to their types, such as player movement keys for the UserObject. Since UserObject is inherited from ActiveObject, VRBridge can easily be extended to provide embodied players with animations. Locations and Waypoints are mechanisms for working with VE space which refer to areas or points and are part of the WorldObject.

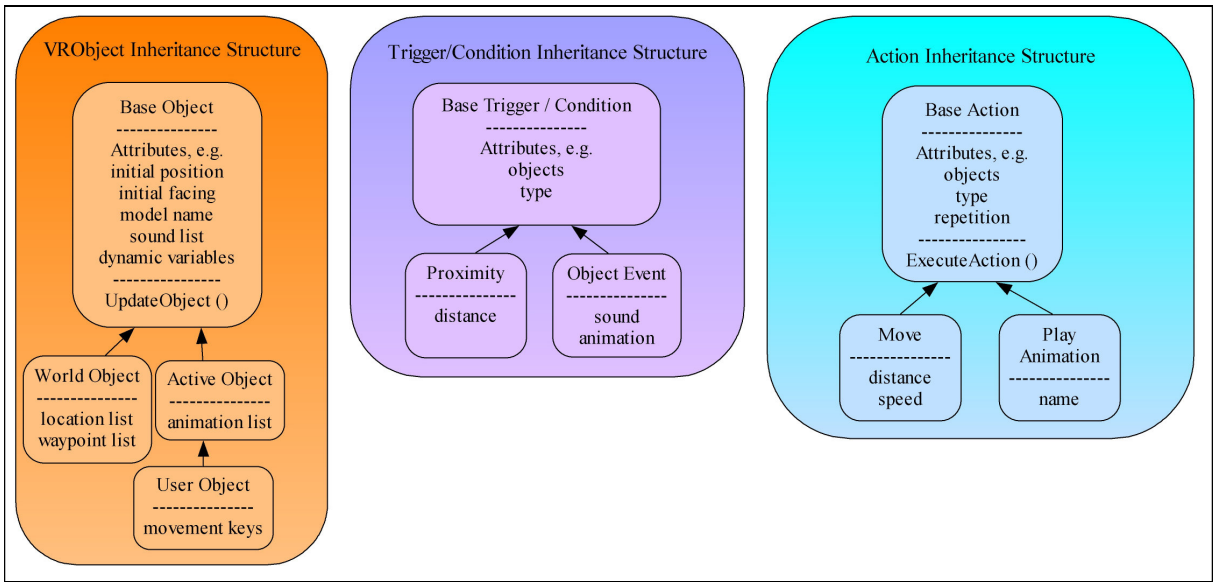


Figure 6.2: The inheritance structures of VRObjects and Triggersets

Triggersets are the mechanism for programming interactions. Each contains a trigger, list of actions and list of conditions. Triggers and conditions are part of the same inheritance structure, while actions are encoded in a separate one. These consist of a Base Trigger/Condition or Action, and inherited classes corresponding to different types, so they can be extended easily. Inherited classes contain pointers to locations, sounds, etc, as the trigger, condition or action requires. Actions also contain a virtual method for action execution.

6.3. Programming Model and Interface for End Users

Here we describe the programming model used for the InterPlay Tool and its interface.

6.3.1. Programming style for end-users

While VE interaction creation above a certain complexity is usually accomplished through programming with high-level languages, the basic interactions are simple when decomposed, e.g. if these conditions apply then this happens. In addition (as shown in Section 4.2.3), most complex interactions can be decomposed into a small set of *building blocks*, which map to universal VE interaction tasks. These can be built up in multiple ways to form interactions. This provides ease of programming interactions and flexibility of use.

We therefore base the interaction programming interface on the event-action paradigm using natural language. Simple events and actions can be specified and combined to form complex interactions. Anything that happens in the system follows from an event. This conforms to our principle of *atomic simplicity*. We adapted Zachmann's scheme for a flexible event-action system (see Section 4.3.2), and designed *Triggersets* for the specification of interactions. We also include the conditions under which an event will execute separately and explicitly in our formalism, as they allow event-action pairs to be customised and extended in context, increasing the expressive power of Zachmann's system. For flexibility of programming, our actions can be triggered by any event (or trigger in our terminology), any action and event can be connected, events can be combined with boolean expressions and an action can be the input for another event. This connects with the guidelines of providing constraints and flexibility, defined in Section 4.3.1. The event-action system provides constraints on the atomic interaction choices that are made, while allowing them to be composed very flexibly. This should provide guidance, while promoting freedom of expression. We describe our Triggersets next.

6.3.2. Triggersets

Triggersets are event-condition-action triads, which are essentially rules for specifying how interactions occur. A Triggerset consists of one triggering event, one or more conditions required for the event to execute, and one or more resulting actions. As actions, triggers and conditions are all separate objects, any combination can be made. Conditions are combined with an implicit boolean AND. As novices have problems deciphering complex combinations of boolean logic (as seen in Section 4.1.3), we avoid using boolean OR as a combination mechanism. Designers can create multiple Triggersets to specify OR conditions. While this increases the time needed to express alternative actions, it reduces the complexity of the programming. Triggers and conditions are similar entities: they both indicate when actions can occur. The difference between them is that a trigger is an instantaneous event that triggers an action, while a condition is a continuing state of the VE that allows the trigger to execute. For example, if a trigger is *Object A starts playing a sound*, the corresponding condition would be *Object A is playing a sound*.

The provided types of triggers and actions must be carefully selected, so that, with a few simple combinable options, complex interactions can be created. Based on our research into typical VR interactions taxonomies (in Section 4.2.3), we distilled the following general requirements of interaction by the player in a VE:

- The player must be able to navigate and change orientation;
- The player must be able to perform tasks by interacting with objects;
- The system must be able to interact with the player.

These requirements focus on the actions of the player in the VE, which are the most essential interactions to consider, as the ability of the player to interact is of primary importance. However, interactions that are not explicitly instigated or performed by the player are also important to the overall experience, as they add to the complexity of the VE world and make it more compelling. We want to allow the end-user to specify various methods for interaction to increase the player's perception of the VE's interactivity.

Our own experience in authoring VEs was considered when defining triggers, conditions and actions. We reviewed a variety of VEs created by our lab to determine the common basic interactions that are typically available. These VEs were created by small teams over a period ranging from one month to two years. They included a VE to educate people about HIV (Brown et al. 2005), a VE to investigate the effect of sound and pictures on presence (Brown et al. 2003), a cultural heritage VE which focussed on storytelling (Ladeira et al. 2004), a museum walkthrough (Hendricks et al. 2003), and small VEs or 3D games created by students in VR courses over several years. This wide variety of VE types gave us a broad base of interactions that might be required by the designer of an interactive VE, and a starting point from which to begin the design of our Triggersets. We were also guided by others' work, e.g. the actions available in Alice (see Section 4.3).

From the work reviewed above, we gathered minimum requirements for actions and triggers. We were especially influenced by Wazlawick et al.'s types of interactions, described in Section 4.2.3.:

- The player and other objects must be able to move and rotate in the VE, and thereby trigger actions. Therefore, for triggers we need proximity, visibility and collision options. For actions, we need movement and rotation options. Based on research about working with 3D space (reviewed in Section 3.1.3), we provided the ability to create Locations and Waypoints; these are 2D areas and 3D points, respectively, which are not visible to the player and can be used for spatial triggers.
- The player must be able to navigate and orient freely in the VE, i.e. not based on Triggerset actions. In general, actions in Triggersets should not apply to the player, who should be in control of her movements.
- The player must have other interactions besides navigation and orientation. Therefore, we need a player input trigger. We follow standard desktop game and VR techniques, and use mouse movement for player orientation in the VE (i.e., looking around). Player input options are keyboard or mouse-button based.

- To create compelling and complex interaction possibilities in the VE world, actions must be able to trigger other actions. Therefore, we need triggers that are based on events, such as sounds and animations. We also need actions that start these events. The ability to change object properties adds to interaction effects, and we therefore need triggers and actions which refer to object properties. For simplicity we only allow the properties of size and visibility to be changed in this way.
- Lastly, we need to consider time in the VE. Providing Triggersets which depend on player interaction to execute creates an implicit design guideline. This is done through fostering awareness of the VE player as an independent and unpredictable entity; and preventing linear time-sequenced actions in the programming interface. However, the ability to coordinate actions of multiple objects for narrative effect (e.g., objects engaged in conversation) provides important expressive qualities. Our research into tool requirements of interaction designers indicated that this was a desired feature (see Section 4.1.3). Therefore, we provide the additional representation of Timelines, which are composite actions that can be triggered like normal actions. We also provide a World Start trigger, for actions to occur as the VE world opens. This allows for the perception of a VE world that exists independently of the player.

Table 6.1 shows the format of the Triggersets, types of triggers, conditions and actions, and their parameters. Triggersets can be activated by the player's actions, or as a consequence of previously triggered actions. As Table 6.1 shows, each trigger or condition can be specified in the affirmative or negative. For example, an Object Event trigger can occur when the event starts or stops (for conditions, this would read as while the event is happening or is not happening). Actions can be repeated over a period of time or for a specified number of iterations. These simple additions increase the expressive power of VE interactions. An example Triggerset is as follows:

if object x enters a distance of 5m from object y (trigger), *when object y is in location c AND object z is performing animation 'dance' AND user is pressing key 'p'* (conditions) *then object y starts sound 'help me' AND object x moves towards the user* (actions).

As can be seen, complex interactions can be programmed using simple atomic parts. There are several kinds of interactions that are not easily created with our system, as we briefly alluded to in Section 4.2.3. We cannot capture uncertainty in the sense of actions that have a percentage chance of happening. We have added timelines to Zachmann's system, as an action in order to deal with the time issue, but they change the nature of the system and are less immediately reactive than the other actions. We also cannot capture complex interactions requiring physics simulations such as particles, or requiring objects to be picked up and associated. Actions cannot be attached to multiple objects, which makes crowds difficult to manage. We also do not have a mechanism for stopping actions, except through interruption. These limitations would need to be addressed to alleviate frustration in novices as they become more expert. However, for the purposes of the

initial prototype, we decided that implementing common interaction possibilities was sufficient for proof of concept. Next we consider the programming interface for Triggersets.

Designer Selects:	Trigger/Condition Types	Num Objects	Parameters	Example
One instantaneous trigger	Proximity	2 objects	Distance	Tom is within 1m of table (cond)
	Collision	2 objects		Tom has just touched table (trig)
	Visibility	2 objects	Field of view, Object(s) viewing	Tom sees table (trig)
AND	Location	1 object	Location	Tom is not in Hall (cond)
	Player Input	player	Key or mouse button	Player is pressing 'x' (cond)
Zero or more prevailing conditions	Object Event	1 object	Type (world start, animation, sound, property, timeline), Select animation, etc	Tom starts animation 'shuffle' (trig)
	Each trigger has a begin / end parameter (e.g., enters or leaves proximity) and each condition has a during / not during parameter (e.g., while or while not in proximity)			
AND	↓			
	Action Types	**Parameters**		**Example**
One or more resulting Actions	Animation	Animation name		Tom plays animation 'nod'
	Sound	Sound name		Tom plays sound 'greeting'
	Timeline	Timeline name		Tom plays timeline 'move to table'
	Rotation	Axis (x, y, z or flickflack, cartwheel, spin) or object to face, Time to rotate		Tom turns to face table over 10 seconds
	Movement	Direction or object to move towards, Distance, Time to move		Tom moves forward 6 metres over 10 seconds
	Change Property	Type (size or visibility)		Tom increases size x2

Table 6.1: Table of Triggerset types, showing how the designer specifies one trigger, zero or more conditions and one or more actions to create a Triggerset.

6.3.3. Programming interface

We wanted to avoid problems of remembering syntax, and include direct manipulation as far as possible. Therefore, we provide dialog boxes and sentence functions with pop-up menus for programming Triggersets; we use dialog boxes as users learn more easily with familiar interfaces, and desktop packages such as spreadsheets and word processors (which use dialog-based input) are ubiquitous. Dialogs also provide constraints which guide users' actions. We designed the interface for programming Triggersets, the InterPlay Tool, according to user interface principles, such as providing visible system state and action alternatives, feedback, and constraints on action. The InterPlay Tool also provides the interface for modifying VE object details. A screenshot is displayed in Figure 6.3.

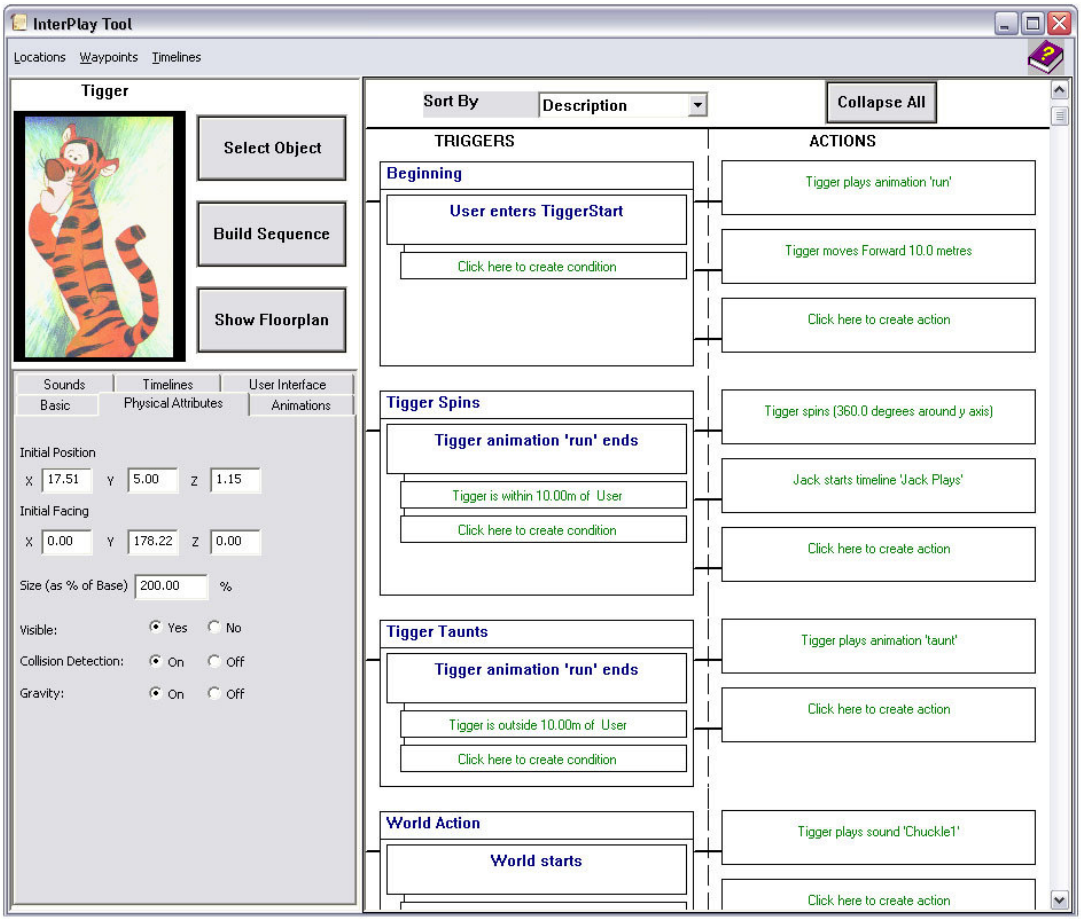


Figure 6.3: User interface for Triggersets and object details. Objects are displayed on the left with their details in tabs organised according to basic and physical attributes, animations, sounds, timelines and player. Triggersets are displayed on the right and can be scrolled, collapsed to show only descriptions, and sorted.

The InterPlay Tool is divided vertically into two parts: the Object Panel and the Triggerset display. Once an object is selected, its details appear on tabs below a picture of its model in the Object Panel. The tabs are

divided by subject-matter and their details can be modified: for example, the Physical Attributes tab displays the 3D coordinates of the object's initial position and facing, its height, size and visibility, whether it responds to gravity, and whether it collides with other objects; the Sounds tab displays a list of all sounds connected to the object and ability to preview them; and the User Interface tab displays details specific to the player, such as the keys for moving the camera, its height and speed of movement. The Object Panel also displays buttons for opening the additional representations.

The Triggerset display was designed to be highly visually organised, while allowing the details to be entered with text. This section of the screen is divided in half, with the left half for Triggers and Conditions, and the right half for Actions. This visually highlights the distinction between triggers and actions. Each Triggerset is contained in its own box, labelled with a user-defined description. These descriptions can be used to identify Triggersets according to personal heuristics. The Triggersets can be sorted alphabetically by description, and collapsed to only show the description. When the Triggersets are collapsed, an overview is obtained, as they can all be viewed on screen at the same time. Gestalt principles of connection and enclosure are used to promote the perception of each Triggerset as a single object made up of trigger, conditions and actions.

The screen always contains one empty Triggerset box, with labels describing the actions to perform a task, e.g. Create Condition. Each Triggerset box contains clearly defined areas for specifying the trigger and conditions. The trigger block is then visually connected to the action blocks. If the designer selects one of these boxes, a list of options for triggers, conditions or actions will appear. Once an option is selected, the corresponding dialog box appears. If the trigger, condition or action has already been specified, options are provided for editing and deleting.

We designed our dialog boxes with user interface principles to be consistent and simple in appearance and functionality, provide informative feedback and easy reversal of actions. For example, the first field in each trigger dialog is the description; the top fields displayed by every trigger, condition or action dialog select the object(s) involved; and the bottom fields displayed on each trigger and condition select the start or end of the event as the trigger. Each required parameter is described simply, e.g. Select Animation. The dialogs use drop-down lists and direct manipulation for parameter selection as far as possible. Where text must be entered (e.g., object distance for proximity), its validity is checked and a clear error message is provided if necessary. The dialog boxes open with sensible defaults as an additional guide. Once a trigger, condition or action has been specified, its description in natural language appears in the corresponding box on the InterPlay screen. Figure 6.4 displays a Proximity Trigger and Rotation Action dialog box.

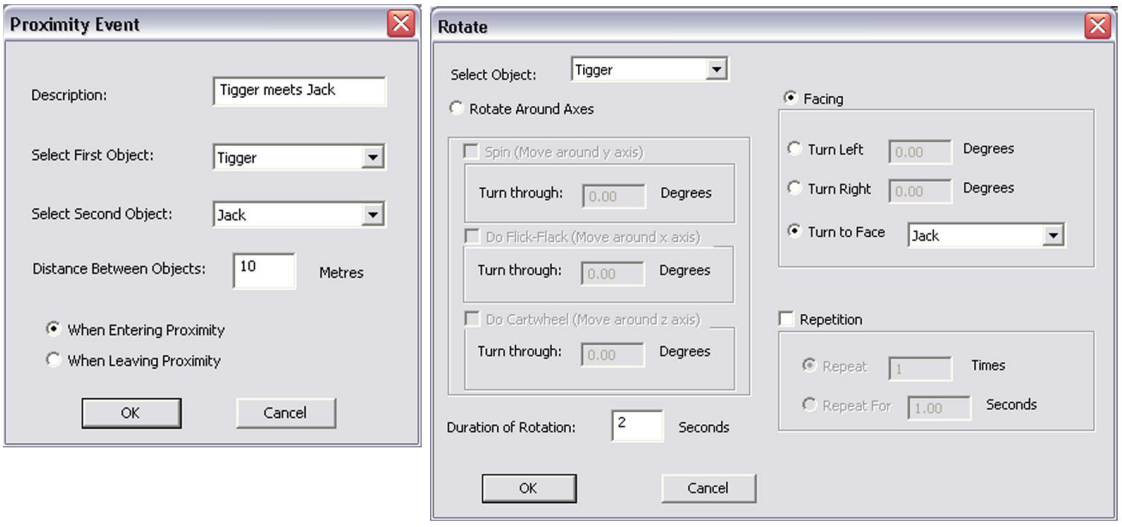


Figure 6.4: Dialog boxes for Proximity Trigger and Rotation Action respectively.

A tension that ran through the system design was supporting novices in creating 3D interactions easily and intuitively, while also supporting learning about the domain. We followed research on intuitive terminology by the designers of *Alice* (see Section 4.3) for our prompts and descriptions. However, we also indicate the alternative, more technical terminology so that users can learn it. E.g., we refer to *movement* instead of *translation*, *size* instead of *scale*, and use various terms for different kinds of *rotation* (i.e., spin, roll, and cartwheel). However, we describe the technical terms where possible, e.g. for *spin*, we include the information that this is a *turn around the y axis*. We also use degrees, rather than the simpler rotations to describe the amount to turn. For more complex actions, such as *movement* and *rotation*, we include simpler, easier to understand redundant options. E.g., for *rotation* we include *turn to face*, *turn left* and *turn right*; and for *movement* we include *move towards*. We address additional issues with technical terms in our scaffolding (e.g., relating degrees to rotations and describing the x, y and z axes).

6.4. Creating the Design Aids: Multiple Representations

Zhang and Norman's (1994) conceptualisation of distributed cognition (discussed in Section 3.1.2) describes how external representations guide and constrain cognitive behaviour. They show how different representations of a task change the way the user thinks about it internally. Novices will use the information conveyed by the representations in constructing their knowledge of how to perform the task of VR interaction authoring. If we examine Zhang's framework for external representation based problem solving (see Figure 3.2), we find several ways to focus our design aids for maximum cognitive benefits.

To recap, Zhang stated that external representations impact the cognitive processes of *lookahead* and biases. Our external representations can help with *lookahead* by adding perceptual signs which point to alternative actions, and indicate their potential consequences. They can also create consistent biases towards performing appropriate actions in interactive 3D. The design aids should attempt to leverage previously learned knowledge to promote problem or domain specific usage. For example, our study on novices (in Chapter 5) showed that they know how to use Floorplans and network diagrams when working in this domain, which suggests that they have internal representations of these forms which are already somewhat effective. We provide representations which correspond to these forms and guide novices towards using them even more effectively. With repeated usage of the representations and VRBridge, useful knowledge will be learned and can be applied to future problem solving. In the discussion below, we will only consider our multiple representations; the benefits and design of scaffolding are discussed in Chapter 8.

6.4.1. Overview and design principles

Our research on diagrammatic external representations and how they can effectively be used (in Section 3.1) shows that they provide direct manipulation for feelings of personal control and allow us to leverage our perceptual advantages. Visualisation extends human cognition through memory extension and spatial reasoning support, as users can view the space and objects of the 3D world continually. Theorists suggest the provision of 2D views to support work in 3D, as they are better used for layout tasks and navigation in 3D is improved with an overview map. Experimental evidence shows that combination views (2D and 3D) are better for orientation and positioning tasks. However, there are problems with occlusion and scene complexity. This result reflects our atomic simplicity principle, and supports our decision to have multiple *separate* representations which are linked.

Having multiple representations of a domain provides a more effective learning space, as different perceptual and cognitive abilities are leveraged with each representation. Therefore, we provide multiple data types: natural language dialogs for interaction programming; 2D diagrams for overview information, and temporal data for events in the VE world. We also provide network data, as novices have been shown to work well with flowcharts, and networks are good for viewing sequencing options. We provide 3D data with our 3D Window, which is good for positioning and orientation while viewing.

We considered several representations for inclusion in VRBridge. Our brief was not to create new representations of the VR design process, but to find existing options which would be easily understood (i.e., no difficult notations to learn) and which could be modified for use in VR. These representations had to help the user to think in non-linear ways and in terms of multiple objects performing interactions. In addition, they had to help the user to consider various aspects of interaction design, such as the use of space, time, narrative sequencing and the independence of the player. In this way, 3D interaction design lends itself to the

use of multiple representations, as it is highly decomposable into various aspects; and this decomposition highlights different features of the problem space.

We created paper prototypes of representation options. After several iterations, where these were discussed and analysed in focus groups, we selected three additional representations to incorporate in VRBridge. These focus on three important characteristics of VR interaction design: usage of space, passage of time and narrative possibilities. This helps to break down the complexity of the design task by focussing on different aspects of it. Because the representations are linked, both explicitly and through the common VE, the designer can also view them as part of a conceptual whole. The representations are as follows:

- *Floorplan*, which provides a 2D overview of the 3D space of the VE, and shows the relative positions and facing of objects;
- *Timelines*, which are both a construction and a visualisation tool allowing designers to skip through lengthy time sequences and place actions accurately in time; and
- *Sequence Diagram* which maps the interaction possibilities of the VE and how it reacts to the player.

These representations provide benefits of a visual system, such as an overview of the data and different perspectives. They help the designer to conceptualise and plan complex relations among the VE objects. They use arbitrary symbols, but provide underlying perceptual cues to make them easier to read. They are also all symbols which are generally well understood as they are used extensively.

Our research indicated that external representations are more useful if people can reproduce them, and that there should be support for tentative actions (see Section 4.1.2). Therefore, we chose simple representations that could be sketched by users who have learned them. Our research also indicated that providing interactivity in diagrams and the ability to follow the flow of information through a diagram helps with cognitive offloading (see Section 3.1.5), as users can actively explore the external representation rather than trying to do this internally. Therefore, the representations also needed to be manipulable, so that users could work with them and control them to some degree.

Figure 6.5 displays a graphical representation of how the various representations in the system are related.

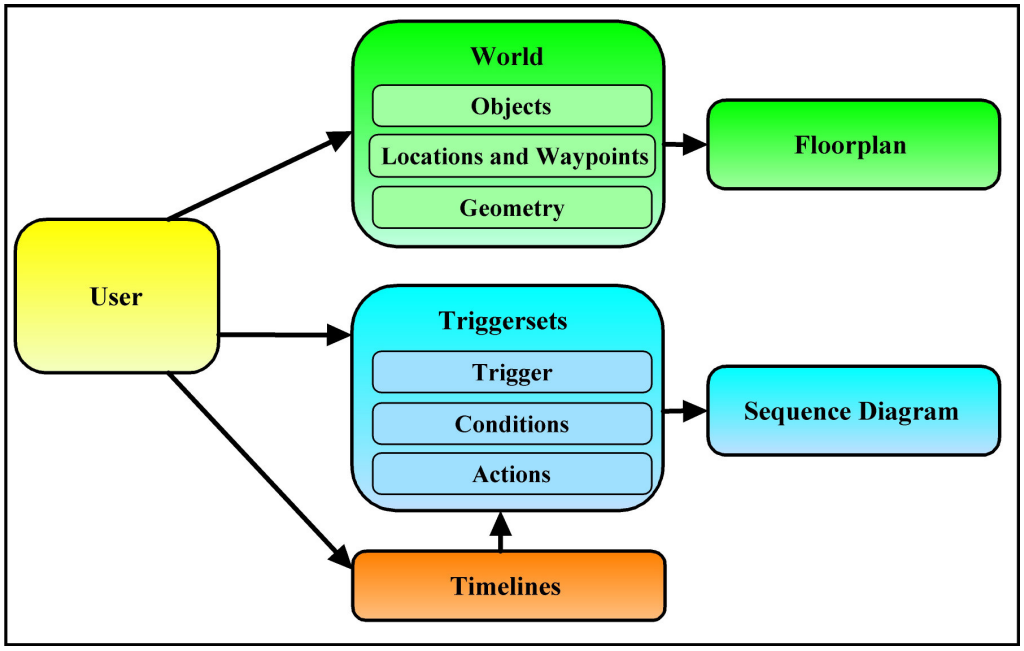


Figure 6.5: How the parts of VRBridge are related. The user enters Triggersets, 3D object details and Timeline information. Timelines may be added to Triggersets. The Sequence Diagram is generated from Triggersets and the Floorplan is generated from the 3D World.

Below, we consider each representation separately, discussing our use of perceptual cues, graphic structure, relationship to the 3D world, and how the representation will assist the user.

6.4.2. Floorplan

The Floorplan was the most obvious additional representation to provide. The usefulness of a continuous overview map for working with 3D was consistently emphasised in our research. Floorplans are used successfully in engineering and architecture to represent space, and similar 2D orthographic projections are used in modelling and CAD packages. Novices tend to understand and use them very easily. People often have problems understanding 3D space, and a simpler 2D representation helps to clarify the effects of acting in 3D. It provides a spatial constraint for the placement and movement of objects. Floorplans can also assist the designer in considering the perceptual opportunities of the 3D world, as described by Fencott (see Section 4.3.3), by providing a simple 2D visualisation of the VE space. In particular, lines of sight, locations, paths and object positions can be easily viewed.

We created the equivalent of a north up map, as these were shown to be more useful than forward up maps for design (Darken and Peterson 2002). We provide a clear indication of the player's position and facing direction to assist orientation. The Floorplan automatically displays the positioning of any object with spatial coordinates and indicates its orientation. The Floorplan is marked with a grid according to the units of the

world, so that the size of the space can be easily interpreted. Waypoints and locations provide landmarks for spatial navigation and divide the space into regions which can be used to describe it. An example Floorplan is displayed in Figure 6.6.

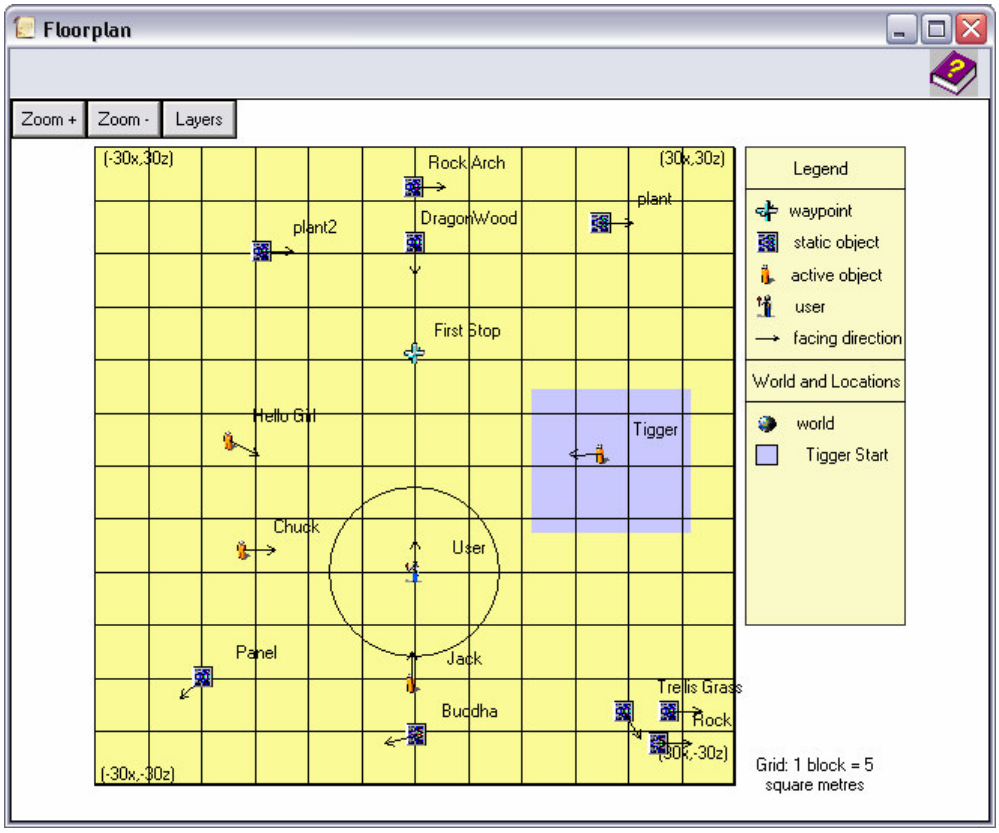


Figure 6.6: Floorplan displaying positions and orientations of objects, waypoints, locations and proximity circles (both locations and proximity circles can be used as trigger regions).

Icons are used to indicate objects and their types on the Floorplan. For instance, there are different icons for basic objects, active objects, the player and waypoints. Locations are indicated by coloured rectangles. Arrows attached to the objects show their orientation, and circles around objects show proximity triggers. A legend indicates the mapping between icons and objects, and also includes an icon for the world, which cannot easily be visualised on the Floorplan. All objects are labelled with their names, except the World and locations which are indicated on the legend.

The Floorplan is divided into layers, so that the displayed information can be made less complex. For instance, the grid, textual elements and active objects are all on separate layers which can be removed. Buttons on the Floorplan also provide the ability to zoom in and out for viewing of more complex spaces.

We do not include the ability to portray multiple levels in a Floorplan (although this could easily be accomplished with a layered approach), as our evaluations did not motivate for this functionality.

The colours for Locations and icons were chosen to be distinctive according to perceptual principles. Gestalt principles were also used for perceptual effectiveness. E.g., the *proximity* principle connects textual labels to their icons, the principle of *connection* connects orientation arrows to the icons, the principles of *enclosure* and *continuity* draw attention to objects in proximity circles, and the principles of *closure* and *enclosure* make locations and the objects they contain obvious. Objects, waypoints and locations can be graphically placed using direct manipulation. Any object's position and facing (i.e., rotation on the y axis) can be changed by selection and dragging.

The spatial aspects of Triggersets are indicated on the Floorplan through proximity, orientation, waypoints and locations, etc. The user gains an overview of the position and orientation of each object and how it will change based on interaction. Any proximity triggers are visible on the Floorplan, which provides with a graphical indication of each trigger's relative area of impact. Any waypoints are also visible on the Floorplan, and become available for objects to move or turn towards in the Triggerset actions. They are also available for access by Proximity, Collision and Visibility triggers and conditions. When locations are created, they can be used in Location triggers and conditions. These mechanisms allow objects to move around the VE world in complex and interesting ways, which may seem natural and random to the player (who cannot view the landmarks used in the interactions).

6.4.3. Timelines

Timelines provide temporal constraints, in the same way that Floorplans provide spatial constraints. They visualise how objects interact during the passage of time, in a way that personal experience of the 3D world and the Floorplan overview cannot. They are a well understood formalism which reduces errors in temporal ordering (Tufte 1993). Because VE interactions are mostly non-deterministic, our Timelines do not represent VE time from start to finish. Instead, they represent parts of the VE where a predictable sequence of actions will happen in known time once they have been started. E.g., if a story is told, the sound file and actions of storyteller and listeners must be coordinated. The Timeline can sequence the storytelling and reactions with precision and efficiency, without the designer having to play the sound file repeatedly. In addition, interactions can be captured, as multiple objects can be placed on the Timeline.

We use simple, non-branching Timelines, as this reduces the complexity of each Timeline. Separate Timelines are used for each event to which the system responds, which allows finer granularity of control and display. In addition, since Timelines only display predictable sequences of events, no actions by the player can be recorded on a Timeline. These two limitations address the tendency to linearity in novice

interaction designers: since timelines are rigid and limited, they are less useful than the Triggersets for expressing interactions; this should limit their use to situations where they have more expressive power, i.e. where sequences must be coordinated. E.g., if a group of actors are conversing and stop when the player approaches, the conversation can be captured on a Timeline. When the player enters proximity, a Triggerset can be defined which will stop the Timeline. Figure 6.7 displays a Timeline from VRBridge.

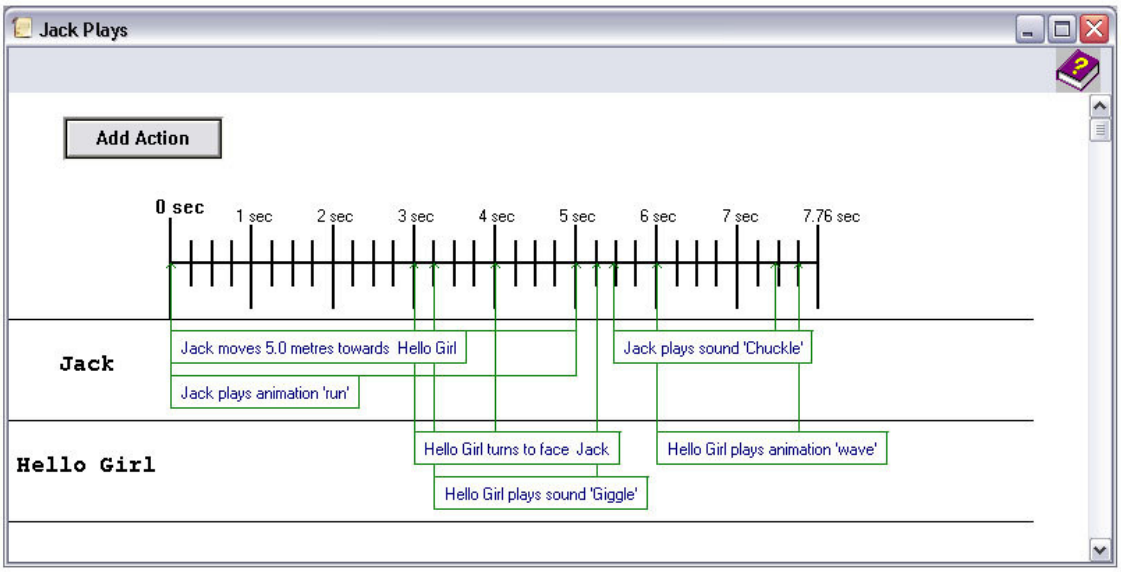


Figure 6.7: Timeline showing objects and their actions, which can be selected to elicit more detail.

When the Timeline is first opened, the designer specifies a name and length, which grows automatically as actions are placed on it. A default length of 5 seconds is provided if nothing is entered. This means that the Timeline opens with a time scale already displayed, thereby providing immediate feedback about its nature.

Timelines provide the benefits of both textual and visual systems, as each action is described in natural language, but they are structured graphically. They are divided vertically, with a defined row for each involved object. Each row contains boxes with descriptions of actions performed by the object. These correspond to the action descriptions shown on the InterPlay screen, which provides consistency between the methods for specifying actions. Gestalt principles were used in the Timeline interface: each action is *enclosed* in a box; lines *connect* the start and end of each action to the time scale. If an action is selected, its box and lines are highlighted to make them more obvious.

To add actions, the user selects the *Add Action* button and specifies a start time (this defaults to the beginning of the Timeline if none is selected). Thereafter, the process is the same as that for adding an action to the Triggersets, which provides consistency. The length of the action on the Timeline is automatically calculated

(e.g., an animation's basic length is read from its file and multiplied by repetitions and speed). Any action can be selected to view more details, or change details. The start time of an action can be changed by dragging it on the Timeline.

Once actions have been placed, the whole Timeline is treated as a single action itself: Timelines can be started like any other action; they are also added to triggers and conditions. Timelines can be created and edited from various points in the VRBridge system. Each is connected to an object, which provides a method of ordering them. Therefore, Timelines of a selected object can be opened from the Timeline tab on the Object Panel. They can also be opened from a menu at the top of the InterPlay screen. Finally, once a Timeline has been added to a Triggerset, it can be opened from that Triggerset.

The Timelines can be used to review how the objects and their actions interact visually. This visual representation of how long an action takes in relation to other activities scaffolds the view of how different interactions work together over time.

6.4.4. Sequence Diagram

For our Sequence Diagrams, we wanted a network diagram that would indicate non-linearity of VR interactions better than the flowcharts used by novices, while still leveraging their ability to read these kinds of diagrams. Since novices naturally created state transition-type diagrams, we decided to create state chart diagrams that indicated states and transitions. A VE may be seen as a complex reactive system which reacts to discrete occurrences. These are the triggers in Triggersets, which can have various consequences depending on time, space and state. The state chart formalism allows us to focus on the actions of the player and how they change VE state. We originally planned to use state chart formalisms for addressing complexity, such as superstates (collections of smaller states). However, when we applied our Sequence Diagrams to typical dynamic VEs, they did not become very complex. Therefore, we did not add functionality to manage display complexity, although the Sequence Diagrams can be extended to include this. We implement geometric zooming to allow parts of a larger diagram to be viewed in detail. The Sequence Diagram does not show an order of events, unless a change in system state makes things possible that were not in other states. This highlights the lack of a predetermined path through a VE for the designer.

The Sequence Diagram is generated directly from the Triggersets entered by the designer. It follows the flow of data through the program, based on the Triggersets and possible player actions. From this, states can be identified where specific interaction possibilities exist (i.e., where player interactions could have consequences). Each state specifies the current VE conditions that allow Triggersets to execute. The states are linked by arrows, which correspond to Triggersets executing, or to possible actions of the player that will

affect the Triggersets. If a Triggerset does not change the VE in a way that makes more interactions available, it leads back to the same state. Figure 6.8 displays a Sequence Diagram from VRBridge.

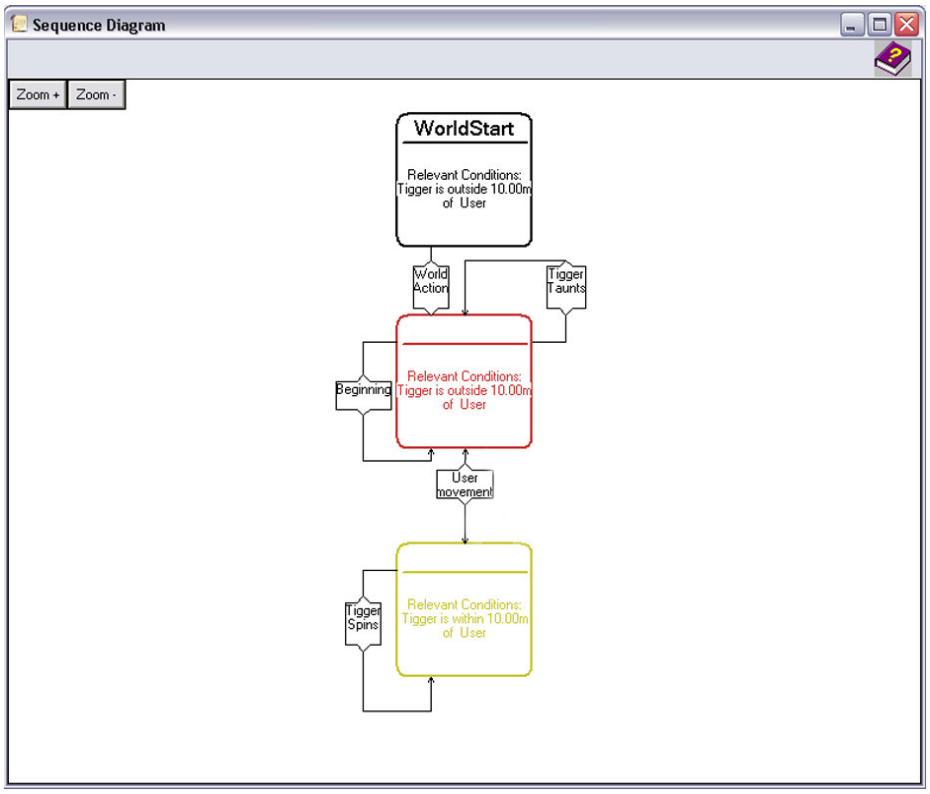


Figure 6.8: Sequence Diagram showing states and links. A state specifies VE conditions where player interactions have consequences, and a link is a Triggerset that can execute or player movement changing state.

The Sequence Diagram provides a visual representation of the Triggersets. In this way it provides redundant information, as the two forms are close to informationally equivalent in the same way as the Floorplan and 3D window are. This in itself can be seen as problem solving support because the provision of redundant information provides more perceptual channels for the designer. The information represented in the Sequence Diagram includes explicit player movement, which is implicit in the Triggersets, as is information about object positions and their effect on Triggerset actions. The potential impact of these details is calculated and visualised by the Sequence Diagram, which should assist the user in understanding connections, and compensate for working memory limitations. E.g., it should help with *lookahead*, by adding perceptual help for the evaluation of action alternatives.

The Sequence Diagram is generated without the 3D Window or Interaction Engine, so the user can visualise how interactions combine without working through alternatives in the 3D world. Therefore, interactions and

their effects on the system must be simulated. The Sequence Diagram can be described as a Simulation Engine that mirrors the Interaction Engine. It uses a recursive algorithm to work out interaction possibilities.

The Sequence Diagram generation algorithm begins in the first frame of the VE by considering if any Triggersets execute on World Start, and simulating that execution. It also considers whether any Triggersets are influenced by player movement, and simulates the player moving towards these goals. For every frame, the algorithm creates a separate set of interactions, depending on whether the player does nothing or acts in a way that will lead to the fulfilment of a trigger or condition. States are defined according to Triggerset conditions, which provide a limit on the number of possible states. When an executing trigger or player movement changes the state of the simulation to fulfil a new condition, a new state is generated. Each transition (Triggerset or player movement) contains pointers to its start and end state. Duplicate states and links are collapsed as the algorithm continues. The algorithm builds a network of possibilities by simulating each frame of the VE in multiple parallel paths. After some experimentation, we set the simulation granularity at 1/10 of a second to calculate interactions. Once all possible states have been visited and all triggers have finished executing, the simulation ends and a diagram is created.

As VE interaction is non-deterministic, it was difficult to validate our algorithm's output. We did this to our satisfaction through extensive testing with members of our lab. We asked testers to play through several VEs authored in VRBridge and document all possible interaction paths. We assigned multiple testers to each VE to maximise analysis completeness. Their results were then compared with our algorithm's output, diversions noted and the algorithm refined. When our algorithm captured 95% of all stated interaction possibilities (and did not produce any impossible paths), we considered its output correct enough to be useful to designers.

As our Sequence Diagram is a directed graph, we considered both graph drawing aesthetics and guidelines for network diagrams in our design. Based on these aesthetics, we programmed our Sequence Diagram to optimise the balance of the following heuristics when drawing line paths: minimum length of connector; minimum number of lines crossed; minimum number of line segments (i.e., corners or twists); and few parallel connectors. We also defined a minimum space between nodes and a minimum enclosing space for connectors. To address Gestalt principles of proximity, our diagrams use auto-positioning, where elements that refer to others in the diagram are automatically positioned close to the element to which they refer.

The Sequence Diagram allows the designer to move conceptually from thinking about individual interactions to thinking about how they sequence together. Thus, it visualises the narrative flow of the VE. The VE narrative sequence usually depends on the player's actions, so it also shows possible reactions of the VE to the player. These reactions must be programmed into the system, so the number of possibilities is limited. By interacting with the Sequence Diagram, designers can step through their interactions. If a designer selects an

arrow, relevant details are retrieved, e.g. the Triggerset corresponding to that arrow. In addition, other arrows with the same details are highlighted. Similarly, if the designer selects a state, details about the conditions comprising the state are provided. In terms of debugging interactions, the Sequence Diagram provides a visual representation of designers' Triggersets, showing their effects, unexpected consequences and which Triggersets never execute. Because the Sequence Diagram is not linear, designers are encouraged to view interactions in a non-linear way. A sparse diagram indicates a lack of interactions provided in the VE: few states and links indicate a lack of nuanced response by the VE to player interactions.

6.5. Linking between the Representations and Triggersets

Once we created all three representations, two important considerations were how to display multiple windows effectively and how to link between them. We chose multiple windows as they provide a simple and effective way to show multiple forms of information: they do not distort the information and can show focus and context simultaneously. We followed Baldonado et al's (2000) guidelines for using multiple representations effectively, such as linking views though slaving and highlighting, using perceptual cues to direct attention and using consistency to simplify the interface (described more fully in Section 3.1.4). VRBridge allows users to open the representations in any configuration for flexible usage. We use perceptual cues to convey the mapping between representations and Triggersets, and to focus the end-user's attention on the correct view/part of view. These include movement, highlighting, anchors (visual objects that are invariants between representations), and coupling through linked or slaved views (i.e., one view changes as another does).

All of our representations are linked in multiple ways. Objects and locations are anchors, as they appear across representations and can be cross-referenced. When an object is selected on the Floorplan, its details open on the InterPlay screen; on the Floorplan, it is highlighted and its proximity triggers shown. When an object is selected on the Object Panel, it is highlighted on the Floorplan. A Triggerset which appears as an arrow in the Sequence Diagram can be opened from there to view its details. The designer can also view the position of objects, waypoints or locations from the Sequence Diagram; the selected object will be highlighted on the Floorplan. Any object's position and facing can be changed using direct manipulation on the Floorplan, or by entering text on the Object Panel. The same is true for waypoints and locations. The details will change in both screens as they are changed in one by the user. In this way, by working through the representations in combination, the designer gains an overview of the interaction design, how the parts interact and combine, and a detailed description of each. We discuss dynamic linking in Chapter 8.

A typical authoring process is as follows. The designer specifies objects and the environment. These do not have to be instantiated in their 3D forms immediately; only their positions, names and approximate details

(such as sounds and animations) need be specified. For programming interactions, the designer enters Triggersets. Time-based sequences of actions are entered using Timelines. The system generates a Floorplan from object positions and locations. It also generates a Sequence Diagram from the Triggersets. As soon as a few Triggersets and object details have been entered, the designer can view the sequences generated by these interactions and the VE space on the Floorplan or in the 3D Window. Thus, the system allows for incremental programming, as the status and consequences of programmed interactions can be checked at any point by examining the representations.

The event-action input mechanism is enriched by the addition of the representations, which help the designer to disentangle and debug interactions. This should allow the user to feel in control and encourage exploration. For example, Triggersets can be set up to introduce contradictions, but these are made apparent by the other representations; a trigger that never fires will not appear on the Sequence Diagram; and the Floorplan will show where objects will collide. In this way, we follow the principle of exploration, which suggests that designers should be allowed to make mistakes so that they learn from them, but that they should be provided with feedback and assisted in recovering from errors.

6.6. Synthesis

In this chapter, we have described the design of VRBridge, focussing on the interface for specifying interactions and the additional representations which support the design process. The representations were designed to complement each other and the 3D Window, so the designer becomes aware of their conceptual relationship: how space, time and sequencing affect each other. Most important are the Sequence Diagram and the Floorplan, as these provide overviews. However, the Timelines are very useful in reducing frustration when synchronising actions. The representations also describe interactions in VE world terms and not in programming terms. This should help the designer to step back from the programming task and think about the player's experience in the VE.

We have tried to support end-users by making the design process intuitive (e.g., by using natural language descriptions and event-based interactions), while at the same time helping them to learn about how experts in this field work, their terminology and their design guidelines. The next step is to evaluate VRBridge to confirm that we have chosen effective representations and that our Triggersets are easy-to-use and useful. Chapter 7 describes this evaluation.

7. Evaluation of VRBridge

After the initial design of VRBridge, we wanted to evaluate it before continuing. We conducted an exploratory study before adding the 3D Window and scaffolding (Winterbottom et al. 2006). This allowed us to focus on the expressive power of the representations, and to design scaffolding based on our observations. Since we wanted to test understanding of the system, we designed a study that required users to analyse existing interactions, as described by the Triggersets and representations, rather than create new ones.

We mainly wanted qualitative information about how participants worked with VRBridge. Specifically, we wanted to study how effectively the representations and Triggersets were used by people without a programming background. We employed observation and structured interviews (Spradley 1998, Banister et al. 1994) to obtain this information. However, we also wanted a quantitative indication of how our design aid of multiple representations affected participants' understanding of VR interaction design. Therefore we divided participants into two groups, one of which received the representations. We provided two tasks to participants and scored their solutions. The first involved describing possible sequences in a set of interactions and the second involved identifying errors in a different set of interactions. We believed that participants with the design aids would produce more accurate solutions in both tasks. In the following sections, we describe the study aims, materials, tasks, participants and procedure.

7.1. Aims

Our specific aims in conducting this study were as follows:

- To evaluate how well participants worked with the multiple representations;
- To evaluate how easily participants understood and worked with VRBridge and the Triggersets;
- To determine the scaffolding that would be most effective in helping users to create interactions.

There were many factors to consider in determining whether we had successfully implemented an effective authoring tool and design aid. These fall into four categories: VR authoring expertise; constructivist design values; system design; and effective external representations. Table 7.1 describes these categories, showing the most important aspects of each. A more detailed description is provided in Appendix B.

7.2. Materials

Here we describe the tasks for the Sequencing and Debugging parts of the study, and the representations provided to the group that were assigned design aids. The VRBridge system and design aids used in this study did not contain all of the functionality of the full system. As stated in Chapter 6, the 3D Window and scaffolding had not been included. In addition, because the focus of this study was analysis rather than

construction, we removed all ability to modify the representations, Triggersets and object details. Some of the functionality described in Chapter 6 was also not available, and was added because of feedback from participants in this study. We have described the system in this way to maintain simplicity, and because none of the changes involved major design work; all were minor interface changes that increased system visibility. The most important differences were that Triggersets could not initially be sorted or collapsed; and less detail was provided on the action descriptions in Triggersets and Timelines (e.g., no angle of rotation).

Category	**Question**	**Aspects To Consider**
VR Authoring Expertise (see Section 4.2.2)	Do VRBridge, Triggersets and representations support expertise in the domain?	Relevant details, player freedom, non-linear thinking, programming, time and space, and composing interactions.
Constructivist Design Principles (see Section 2.4)	Do VRBridge, the Triggersets and design aids embody Constructivist design values?	Simplicity, multiplicity, exploration, control and reflection.
System and Interface Design (see Table 4.1)	Have we designed an effective system and interface for end-users?	Clear mapping, visibility, constraints, flexibility, error-handing, feedback and terminology.
Effective External Representations (see Table 3.1)	Have we created effective external representations for our design aids?	Distributed cognition (*lookahead* and biases), cognitive offloading (working memory and perception), re-representation, and temporal and spatial constraints.

Table 7.1: Aspects to consider in evaluating the successful design of VRBridge and the multiple representations. These fall into four categories: VR authoring expertise; Constructivist design values; System and Interface Design; and External Representations.

7.2.1. Sequencing task

We provided participants with jumbled Triggersets and asked them to describe sequences of interactions that could happen in a VE based on the Triggersets and object details. Based on our experiences in VR authoring and our universal actions described in Section 6.3.2, we defined the following interactions to include:

- Player input triggering an effect and Triggersets requiring player movement and orientation;
- Actions triggered by previous interactions, e.g. movement of VE objects;
- VE conditions affecting trigger execution.

We designed a simple VE entitled *Bouncer Example*. The space consists of three locations: the Entrance where the player is placed initially; the Waiting Room which opens off the Entrance; and the Inner Chamber

which also opens off the Entrance and is locked. The VE contains a Bouncer (or guard), Door (between the Entrance and Inner Chamber), Bell, Suitcase and Chalice (invisible initially). The goal is to distract the Bouncer away from the Door by ringing the Bell in the Waiting Room, so that the player can open the Door to the Inner Chamber, open the Suitcase and get the Chalice. We provided all participants with a description of the space and positioning of locations. The positioning and facing of all objects was available through the VRBridge Object Panel. Triggersets describing the possible interactions were created with meaningful descriptions; some could not execute without others having already been triggered. Table 7.2 displays the *Bouncer Example* Triggersets, in the order in which they appeared on the InterPlay screen.

Description	**Trigger**	**Condition(s)**	**Action(s)**
No Entry	Player presses 'x'	Player is within 0.5m of Door AND Bouncer is within 1m of Door	Bouncer plays sound 'no entry' AND Bouncer plays animation 'lift hand'
Entry Allowed	Player presses 'x'	Player is within 0.5m of Door AND Bouncer is outside 1m of Door	Door plays animation 'open'
World End	Player presses 'x'	Player is within 0.5 m of Suitcase	World plays sound 'you win' AND Suitcase plays animation 'open' AND Chalice becomes visible
Bell Ring Effect	Bell sound 'ring' ends	Bouncer is within 1m of Door	Bouncer turns right 90 degrees AND Bouncer moves forward 6 metres
Bell Ring	Player presses 'x'	Player is in Waiting Room	Bell plays sound 'ring'
Close Door	Bouncer moves within 1m of Door		Bouncer turns right 90 degrees AND Door plays animation 'close'
Bouncer at Bell	Bouncer moves within 1m of Bell		Bouncer plays animation 'scratch head'
Bouncer Returns	Bouncer animation 'scratch head' ends		Bouncer turns right 180 degrees AND Bouncer moves forward 6 metres

Table 7.2: Triggersets provided for the *Bouncer Example*. The columns show the Triggerset descriptions, triggers, conditions and actions respectively.

The representations for the *Bouncer Example* were a Floorplan, a Sequence diagram, and two Timelines: *Bouncer Move* and *World End*. *Bouncer Move* Timeline (see Figure 7.1) is triggered by the *Bell Ring Effect* Triggerset and replaces the *Bouncer at Bell* and *Bouncer Returns* Triggersets. *World End* replaces the actions in the *World End* Triggerset. The Sequence Diagram graphically displays the connections between Triggersets, player movements and VE state (see Figure 7.2). The Floorplan graphically displays the positions, names, facing and proximity circles of objects, locations, and the VE space (see Figure 7.3).

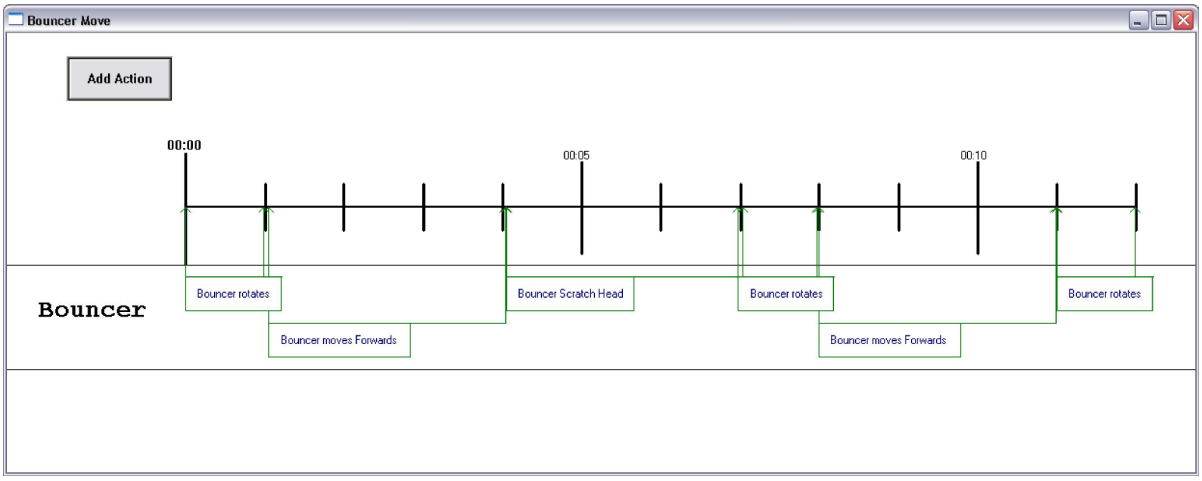


Figure 7.1: *Bouncer Move* Timeline, showing sequenced actions where the Bouncer moves away from the Door, performs an action at the Bell and returns. This replaces the *Bouncer at Bell* and *Bouncer Returns* Triggersets.

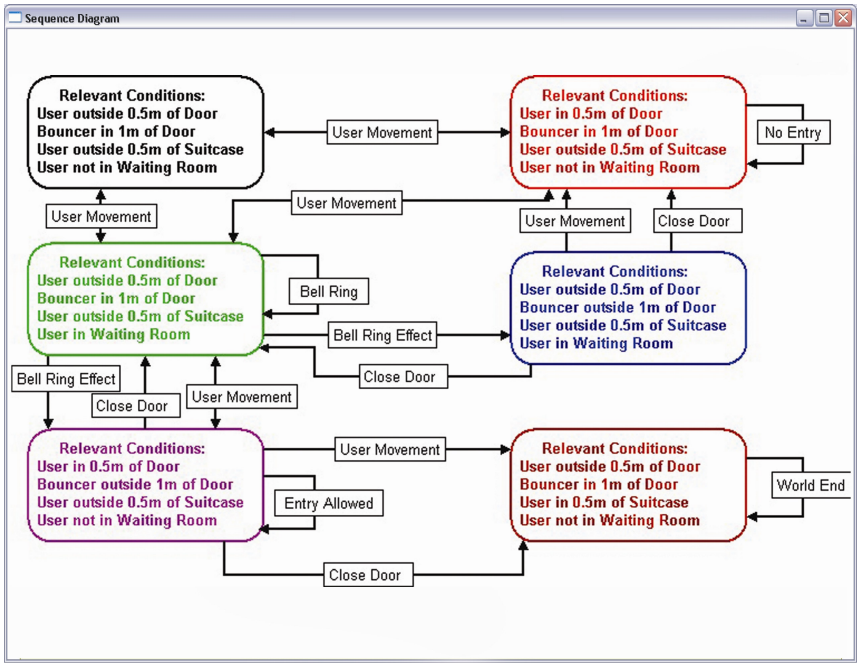


Figure 7.2: Sequence Diagram of Bouncer Example, showing conditions in each state of the VE, and how the Triggersets and User Movements change state. Its layout has been manually adjusted.

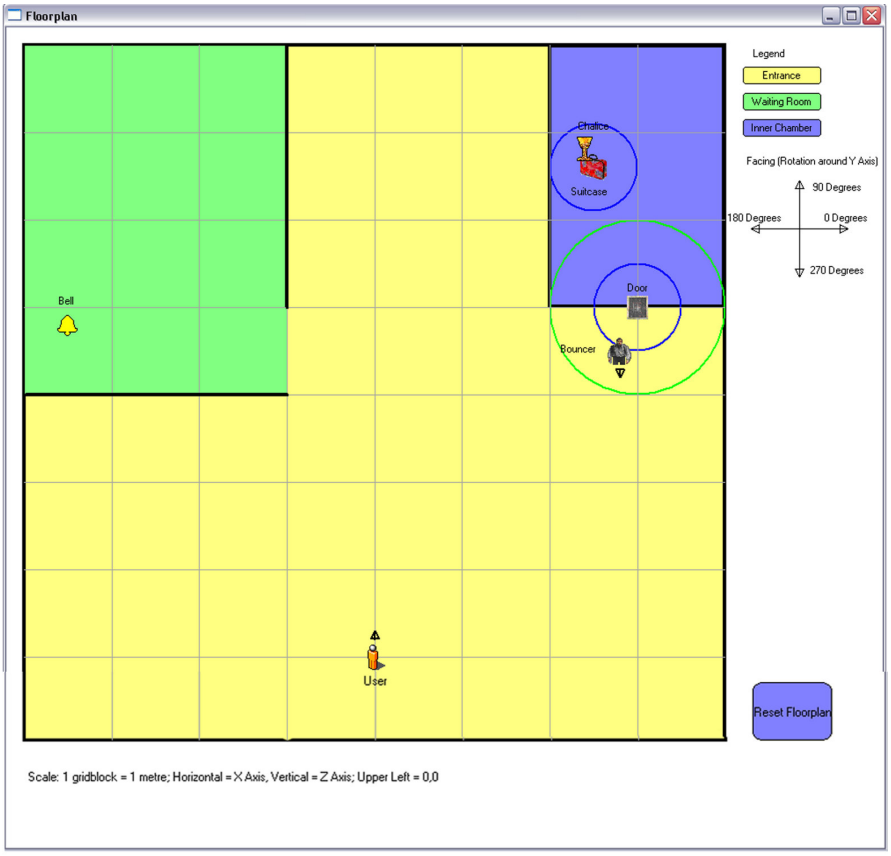


Figure 7.3: Floorplan of *Bouncer Example*. It shows the positions and names of all objects, the orientation of active objects, the locations of the VE, proximity circles and a grid with the VE scale.

We hypothesised that the Timelines would make sequencing interactions easier to understand than the equivalent Triggersets, as they explicitly show how actions relate to each other in time. The Floorplan would make the consequences of interactions in the *Bouncer Example* easier to understand as it visually displays the space in which they occur, and object position and facing. It also makes the perceptual opportunities of the VE clear, e.g., it shows that the Chalice and Suitcase are in the locked location and the Bouncer is directly in front of the Door. Participants can also use the Floorplan to visualise the Bouncer's movements. We hypothesised that the Sequence Diagram would make sequences of interactions in the *Bouncer Example* easier to identify, as it explicitly indicates how the player can interact with the VE. The network formalism should help participants understand the non-linear structure and follow narrative paths, e.g., participants can follow the arrows and consult the corresponding Triggersets to understand how the interactions fit together.

7.2.2. Debugging task

We provided participants with preconfigured Triggersets containing known errors. We informed participants of the intended effects of the Triggersets and asked them to identify the errors. To guide the introduction of

errors into the Triggersets, we identified a taxonomy of typical mistakes in VR authoring. Most mistakes in our experience (and study of novices from Chapter 5) fall into one or more of four general categories: timing, spatial, sequencing & logical and implicit assumptions. These are described in Table 7.3.

Category	**Description**	**Example**
Timing	Errors arising from the time it takes the player or other objects to complete actions	Player does not have enough time to move through a door before it closes
Spatial	Errors in the way space is understood, in terms of orientation and location of objects	An object is set to turn the wrong way and therefore moves in the wrong direction
Sequencing & Logical	Errors in the ordering of the Triggersets and the way that they interact	A trigger never executes because it is not accessed by other triggers
Implicit Assumption	Forgetting to state all behaviour explicitly and provide accurate details	An actor is assumed to be facing the player

Table 7.3: Typical Interaction Programming Mistakes: timing, spatial, sequencing/logical and implicit assumption. The columns show a description and an example for each category of mistake.

Mistakes often fall into more than one category; this suggests that having multiple perspectives on the debugging process will make it easier to find them. If a mistake is not found by examining one aspect of the design, it might be found by examining others. Timing is the most difficult error-type to detect: it is almost always a player-related problem, which means that the designer has to execute and step through the VE to understand it. Because this experiment did not include a 3D Window, we did not include timing errors.

We designed a VE entitled *Jail Example*. The space consists of four locations: the Yard where the player and active objects are placed; the Entrance which opens off the Yard and is locked; and the Cloakroom and Freedom which open off the Entrance and are locked. The VE contains a Jailer, Sandra, three Doors, a Push-Button and a Chalice. The Push-Button is in the Cloakroom and opens the OuterDoor to Freedom. The Chalice is near Sandra in the Entrance. The goal is to ask Sandra to distract the Jailer, so that the player can escape the Yard and enter Freedom. We provided participants with a description of the VE space, its objects and locations. We created Triggersets describing the possible interactions, and introduced four mistakes into them, which prevented some from executing correctly. Table 7.4 displays the Triggersets, in the order in which they appeared on the InterPlay screen. The descriptions of problematic Triggersets are highlighted.

Description	**Trigger**	**Condition(s)**	**Action(s)**
Locked In	Player moves within 1m of Jailer	Player is within 0.5m of YardDoor AND	Jailer plays sound ‘you’re stuck here’ AND

Description	**Trigger**	**Condition(s)**	**Action(s)**
		Jailer is within 1m of YardDoor	Jailer plays animation ‘glare’
Get Out	Player presses ‘x’	Player is within 0.5m of YardDoor AND Jailer is outside 1m of YardDoor	YardDoor plays animation ‘open’
Ask for Help	Player presses ‘x’	Player is within 0.5m of Sandra	Player plays sound ‘help me’
Sandra says No	Player sound ‘help me’ ends	Player is within 0.5m of Sandra	Sandra plays sound ‘no way’
Sandra says Yes	Player sound ‘help me’ ends	Player is within 0.5m of Sandra AND Player is pressing ‘z’	Sandra plays sound ‘for the chalice, I will’ AND Chalice disappears
Sandra Move	Sandra sound ‘for the chalice, I will’ ends		Sandra turns left 90 degrees AND Sandra moves forward 7m AND Sandra plays sound ‘whistle’
Sandra Calls Jailer	Sandra sound ‘whistle’ ends		Sandra turns to face Jailer AND Sandra plays sound ‘come here, big boy’
Jailer Moves	Sandra sound ‘come here, big boy’ ends		Jailer turns to face Sandra AND Jailer moves 5m towards Sandra
Sandra Strips	Jailer moves within 1m of Sandra		Sandra plays animation ‘strip’ AND Jailer plays animation ‘leer’
Caught	Jailer moves within 1m of Player		Jailer plays animation ‘grab’ AND Jailer plays sound ‘I’ve got you’
Jailer Return	Sandra animation ‘strip’ ends		Jailer turns to face YardDoor AND Jailer moves 5 metres to YardDoor
Door Close	Jailer moves within 1m of YardDoor		YardDoor plays animation ‘close’
Press Button	Player presses ‘x’	Player is within 1m of Push-Button	Push-Button plays sound ‘click’ AND OuterDoor plays animation ‘open’
Escape	Player enters Freedom		Player plays sound ‘I’m free’

Table 7.4: The Triggersets provided for the *Jail Example*. The columns show the Triggerset descriptions, triggers, conditions and actions. The Triggersets containing errors are highlighted.

Although we only introduced four mistakes into the Triggersets, the highlighting in Table 7.4 shows that many more Triggersets are affected, as they depend on each other. The errors are identified below:

- **JailerMove**: the Jailer does not move far enough in Triggerset *Jailer Moves*, which prevents the proximity trigger for *Sandra Strips* from executing. Therefore, the *Sandra Strips*, *Jailer Return* and *Door Close* Triggersets cannot execute. This is primarily an example of a *spatial* error. However, it has elements of *implicit assumption* as the movement has not been accurately specified, and *timing* as the problem is with the length, not the form of the action.
- **OpenFreedom**: no Triggersets open the InnerDoor to the Cloakroom. Therefore the player cannot reach the Push-Button which opens the OuterDoor and so the *Press Button* and *Escape* Triggersets cannot execute. This is an example of *implicit assumption* and *spatial* errors. There is no way to open the InnerDoor, and the space of the VE where all the doors are locked has not been considered. It also has elements of *sequencing & logical* errors, as the Triggerset dependencies have not been considered.
- **TriggerConflict1**: the *Caught* and *Locked In* Triggersets both execute when the player moves within 1m of the Jailer or vice versa near the YardDoor, as they share the same trigger (with the exception that *Caught* executes without *Locked In* when the Jailer is away from the YardDoor). This is an example of a *sequencing/logical* error, as the designer has not considered the equivalence of the player approaching the Jailer and the Jailer approaching the player; and has not added conditions to the *Caught* Triggerset to prevent it from conflicting with *Locked In*. This error also has elements of *spatial* and *timing* errors, as the designer has not considered that the player can approach the Jailer at any time.
- **TriggerConflict2**: the *Sandra Yes* and *Sandra No* Triggersets both execute and conflict, unless the player does not press 'z', in which case only *Sandra No* will execute. This is an example of a *sequencing & logical* and *implicit assumption* error.

The representations provided for the *Jail Example* were a Timeline (see Figure 7.4), a Floorplan (see Figure 7.5) and a Sequence Diagram (see Figure 7.6). The Timeline, *Sandra Distracts Jailer*, is triggered by the *Sandra says Yes* Triggerset and replaces *Sandra Move*, *Sandra calls Jailer* and *Jailer Moves*. We hypothesised that the Timeline would make debugging easier by showing connecting actions and where these end, thus helping the designer to identify where errors occur. We hypothesised that the Floorplan would make debugging easier by visually displaying the positions of objects like the Push-Button, and the distance between the Jailer and Sandra. We hypothesised that the Sequence Diagram would make debugging easier by revealing which Triggersets never execute, e.g. *Jailer Return*. This result can then be examined to determine why the break occurs. It can also be seen that *Sandra says Yes* and *Sandra says No*, and *Caught* and *Locked In* both execute from the same state at times, which should hint at their conflict.

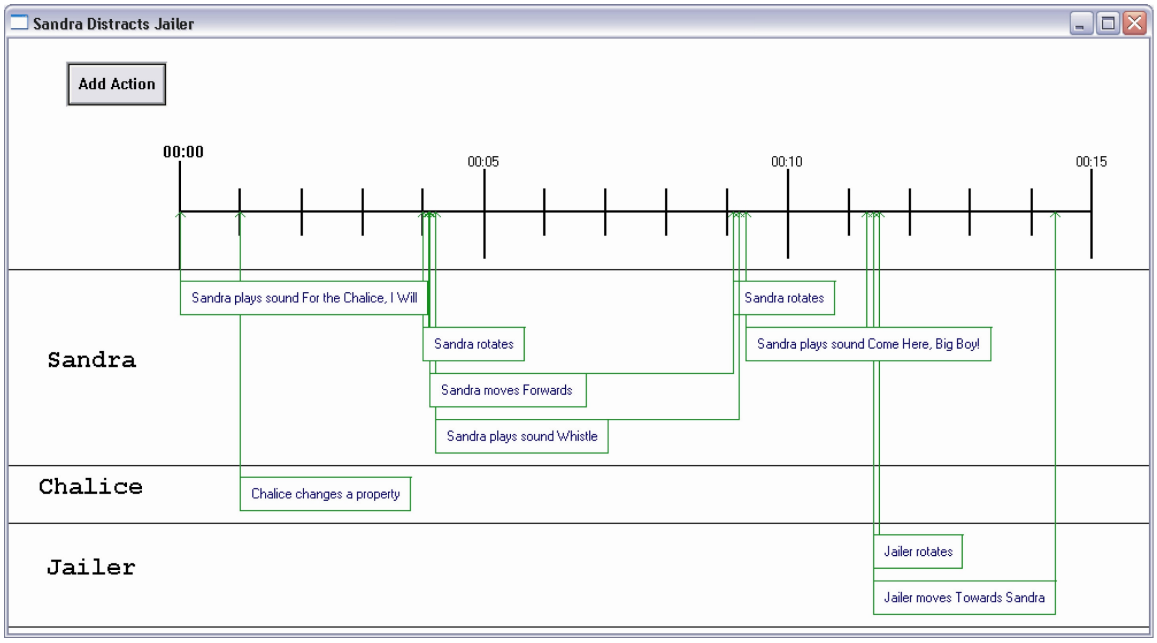


Figure 7.4: *Sandra Distracts Jailer* Timeline from *Jail Example*. This shows sequenced actions where Sandra agrees to distract the Jailer, takes the Chalice, moves away, calls the Jailer, and the Jailer moves to her. It replaces the *Sandra Move*, *Sandra calls Jailer* and *Jailer Moves* Triggersets.

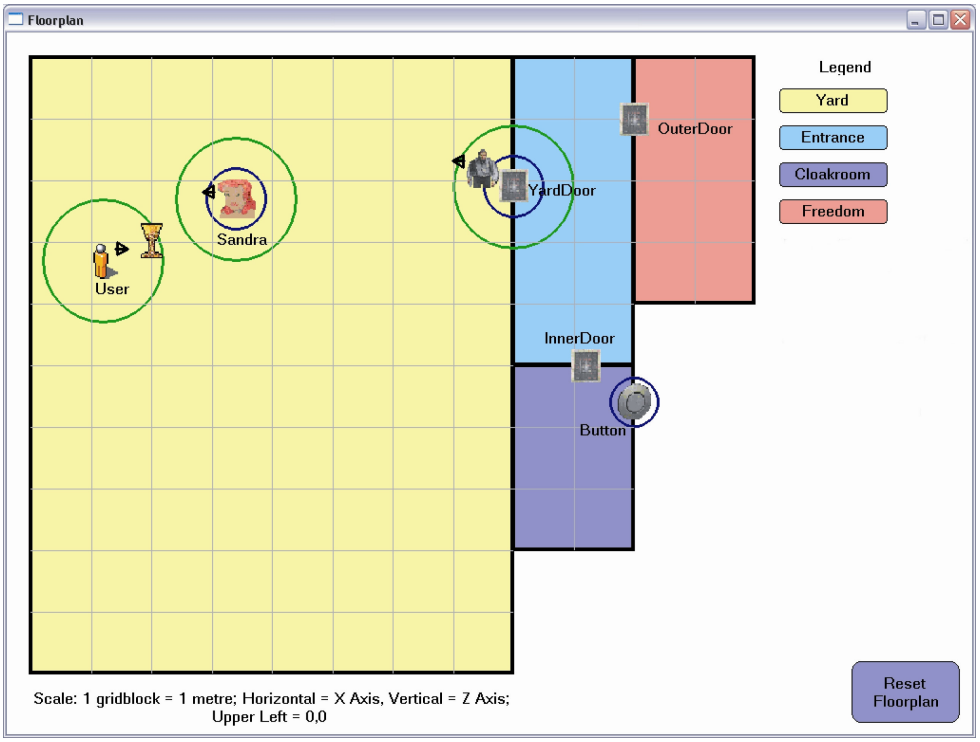


Figure 7.5: Floorplan of *Jail Example*. It shows the positions of all objects in the VE, the orientation of active objects, the locations, proximity circles and a grid with the VE scale

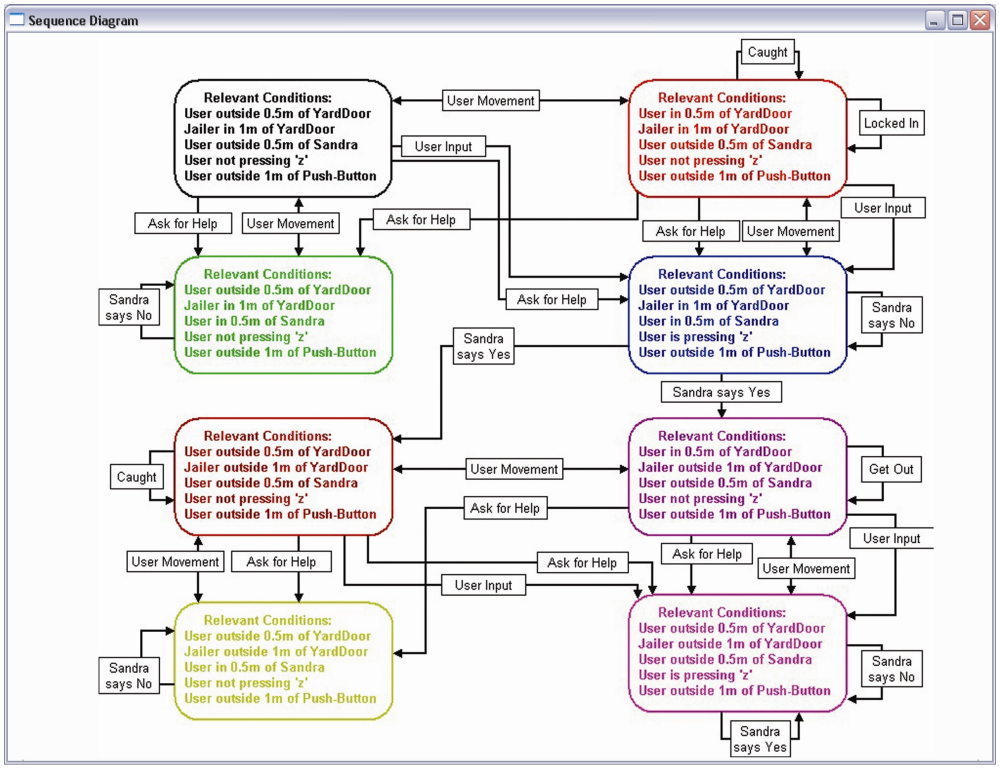


Figure 7.6: Sequence Diagram of *Jail Example*, showing conditions in each state of the VE, and how the Triggersets and player movements change state. Its layout has been manually adjusted.

7.3. Participants

We recruited eleven students from various disciplines through advertisements on university notice boards. All participants had graduate degrees or equivalent experience, and worked in a wide range of disciplines: graphic design, architecture, education, astrophysics and philosophy. All participants were experienced with computers, as we did not want lack of computer familiarity to be an extraneous variable. Four of the participants had used CAD systems, and one had used 3DStudioMax. Four of the participants had some programming experience, but this was at a basic level using Flash, Visual Basic or Matlab. The sample included six men and five women between the ages of 23 and 35. They were volunteers and paid a small amount for their time. We made sure that those with programming and CAD experience were equally divided between groups. Apart from this restriction, the groups were randomly assigned. One was given the VRBridge system without representations (Basic group) and the other was given the VRBridge system with representations (Representation group).

7.4. Procedure

Each participant worked individually and was then interviewed. Before introducing the tasks, we described VR, dynamic interaction and the role of the player. The VRBridge system was then described and demonstrated, as were the representations for those who received design aids. We then gathered demographic details, and gave participants the instructions and VE descriptions for the Sequencing part of the study. Both groups were given a description of the VE space and its coordinate system, coordinates of VE locations, and a list of included objects. They then had 20 minutes to examine the material. They were allowed to make notes during this time. After 20 minutes they were asked to write down what could happen in the VE, indicating dependencies in how Triggersets might execute. Thereafter, we gave participants instructions and VE descriptions for the Debugging part of the study. We described what was supposed to happen in the VE, and indicated that there were errors in the Triggersets which would stop this. Participants were asked to identify errors and had 20 minutes to examine the Triggersets. The experimenter was present at all times to observe participants.

After the tasks we interviewed participants individually. This was structured according to specific questions, but was conversational in form. We began by discussing participants' written output, where they were asked to explain their solutions and discuss their experience of working with Triggersets. Those who had received design aids were asked questions about the representations: their relative usefulness; how they interacted and how they might be improved. The questions used to guide the interview are presented in Table 7.5.

Group	Question
Basic and Representation Groups	Explain what sequences of events can happen in the first environment, using your notes.
	Explain what mistakes you found in the second environment, using your notes.
	Did you understand everything? How difficult was it to find sequences and errors?
	How did you find working with the Triggersets? Were they understandable?
	How would your improve the interface and make the Triggersets easier to understand?
Representation Group only	Did you find it useful to have the representations available? How?
	Rank the representations and explain your ranking?
	How did the representations compare against each other in terms of usefulness?
	Did you use the representations in different ways or for different problems?
	How would you improve each representation and make it more useful?

Table 7.5: Questions guiding the structured interview during the evaluation of VRBridge and the design aids.

We analysed the interview results, guided by our original aims, to gain qualitative information about how participants worked with VRBridge. We also scored sequencing accuracy for the first part of the study, and

counted discovered errors for the second. We developed a code to score the sequence descriptions: participants scored on correct sequencing; and awareness of interaction branches, Triggerset dependencies, object locations and how these might change. We introduced redundancy in the code to allow for participants stating information in different ways. E.g., participants scored up to four points on the *Bell Ring Effect* Triggerset for mentioning various details corresponding to the trigger, conditions and actions. Two independent markers coded the scripts to ensure that the marking was unbiased. No discrepancies were found. Table 7.6 displays the code. Each row scores one point, except GS, which scores two.

Code	Triggerset	Description
GS	All Triggersets	Correct general sequences
NE	No Entry	If Bouncer at Door, cannot enter
BR1	Bell Ring	Player must go to Waiting Room
BR2	Bell Ring	Player must press x in Waiting Room to ring bell
BE1	Bell Ring Effect	Only works if Bouncer is at Door when Bell stops ringing
BE2	Bell Ring Effect	Moves Bouncer away from the Door
BE3	Bell Ring Effect / Bouncer at Bell	Bouncer moves towards the Bell and scratches his head
BE4	Bell Ring Effect / Bouncer Returns	Bouncer moves back to Door
EA1	Entry Allowed	Player must be close to Door and press x to open door
EA2	Entry Allowed	This only works when the Bounder is not at the Door
WE1	World End	Player must get to Inner Chamber to trigger World End
WE2	World End	Player can only get to Inner Chamber if Door is open and Bouncer away
WE3	World End	Player must press x near the Suitcase for the World End to happen
WE4	World End	If Player presses x, World plays a sound, Suitcase opens and Chalice appears
CD1	Close Door	When Bouncer returns, he closes the Door which accesses the Inner Chamber
CD2	Close Door	Player is trapped in Inner Chamber, if there when the Door closes

Table 7.6: The code used to score the Sequence part of the study. Code numbers refer to particular Triggersets.

7.5. Results

Here we present quantitative results from the study, using our observations to explain certain outcomes. The interview and observation results will be described as we discuss the results against our aims in Section 7.6. Where we refer to individual participants, we use the letters B and R to denote those from the Basic and Representation groups respectively. Because of the small size of the sample, we only calculated descriptive statistics from the quantitative data. The four participants with programming ability performed the best in

their groups (B3, B5, R2, R6), with the participant with 3DStudioMax experience also performing well (R4). There were no statistically significant differences between men and women.

7.5.1. Part 1: Sequencing

The Representation group achieved a 74.5% average on correctly finding sequences in the first part of the study, while the Basic group only achieved 56.4%. Table 7.7 indicates the total scores of the participants for this part of the study, and their totals for each Triggerset.

ID	GS Max 2	NE Max 1	BR Max 2	BE Max 4	EA Max 2	WE Max 4	CD Max 2	Total Max 17
B1	0	1	2	1	2	2	0	8
B2	1	1	2	1	1	2	1	9
B3	1	1	2	2	2	1	1	10
B4	1	0	2	2	0	1	0	6
B5	2	1	2	3	2	4	1	15
R1	2	0	2	3	2	3	1	13
R2	2	0	2	3	2	3	2	14
R3	2	1	2	3	2	2	1	12
R4	2	1	2	3	2	2	2	14
R5	1	0	1	3	1	3	1	10
R6	2	1	2	3	2	3	0	13

Table 7.7: Scores of participants for Sequencing part of study, with columns showing totals for each Triggerset. The participants with an ID containing B did not have representations, and those with an R had representations.

When we examine Table 7.7, several noteworthy details emerge. Two points were given for a correct overall sequence (GS). Of the Representation group, all but one participant gained full points, compared to only one of the Basic group. This participant (B5) performed better on the Sequencing part of the study than anyone else. This is interesting because of her background (astrophysics) and how she performed the task. She has extensive mathematical experience and has programmed in Matlab. This allowed her to understand the coordinate system and the mathematics of the interactions very well, e.g., how translations and rotations change object position. She also drew an accurate Floorplan and timed Sequence Diagram of her own. Therefore, she effectively re-created the diagrams given to the Representation group.

The Representation group was more likely to leave out details that were not part of a meaningful sequence, indicating that they were more aware of how Triggersets fitted into an overall picture. The Basic group

generally wrote down everything that they noticed. E.g., if the player approaches the Door while the Bouncer is nearby, the Bouncer says 'No Entry' (NE). This Triggerset does not fit into a sequence, as it does not change the VE state. Only half the Representation group noted it, as opposed to all but one of the Basic group. CD1 and WE4 show similar results.

The constraints provided by conditions on triggers executing were also noticed more effectively by the Representation group. E.g., the *World End* Triggerset can only be executed from the Inner Chamber (WE1), which the player can only enter when the Bouncer is away from the Door (WE2). These constraints are not explicitly stated in the Triggerset. The first is a function of the Suitcase's location and the second is a condition on the *Entry Allowed* Triggerset, on which *World End* depends. Most of the Representation group noticed these facts, compared to only one of the Basic group. This was B5, whose skills were noted earlier. EA2 also scores a point for noticing that the player can only access the Inner Chamber when the Bouncer is away. Because the constraint was explicitly stated here (as a condition), half the Basic group noticed this.

Participants, in general, seemed to have difficulty noticing consequences of executing triggers when they were implicit. E.g., the Triggersets specify that if the Door is open when the Bounder returns to it, he shuts it. If the player is in the Inner Chamber at that time, she is trapped there, thus providing an end condition (or at least stalemate) for the VE. This possibility is not explicitly stated, as it relies on player movement (CD2). Only two participants, both from the Representation group, noticed this possibility.

In general, the Representation group did better in understanding the effects of translations and rotations, perhaps because these actions were grouped on a Timeline. E.g., in the *Bell Ring Effect* Triggerset (BE2), the entire Representation group and only two of the Basic group noted that the Bouncer moves away from the Door. However, where more specific details were required, neither group did particularly well. In the same Triggerset, participants were much less likely to know where he moved to (BE3 and BE4). The Representation group were much better at noting locations (BR1, WE1, EA2). The Basic group was only able to articulate these when they were explicitly stated in the Triggersets.

7.5.2. Part 2: Debugging

Table 7.8 displays the results for the second part of the study. The headings, *JailerMove*, etc., refer to errors (described in Section 7.2.2). The percentages indicate the proportion of each group that noted each error. The Representation group detected more than twice as many errors as the Basic group. Overall, the percentage of errors noted was not high. However, participants were limited to 20 minutes to familiarise themselves with the VE and find errors. They were also unable to experience the VE dynamically in 3D, which would have made errors apparent. Only one participant (Representation group) found *JailerMove* and *TriggerConflict1*, compared to five Representation participants and two Basic participants finding *TriggerConflict2* (which was

the simplest mistake in terms of the number of error types that it involved). Although in all cases the Representation group found more errors, both groups had more problems with those mistakes that combined multiple error types and required drilling down to find details. The representations helped here to some extent. The *JailerMove* error can only be found by closely examining the VE space and the details of the Jailer's movement. The *TriggerConflict1* error requires an understanding of the VE space and the player's freedom of movement.

	JailerMove	OpenFreedom	TriggerConflict1	TriggerConflict2	Total
Representation	**17%**	**33%**	**17%**	**83%**	**37.5%**
Basic	**0%**	**20%**	**0%**	**40%**	**15%**

Table 7.8: Percentage of each group that detected errors in the Triggersets.

Overall, only two of the Basic group found any errors. Only one member of the Representation group did not find any errors. The maximum number found was three (Representation group). B5 found two errors, and is the outlier described previously. Without her results, the Basic group would only have found one error.

7.6. Discussion

Our general aims in this study were to evaluate how well participants worked with the design aid of multiple representations, how easily they understood and worked with VRBridge and the Triggersets, and to determine the scaffolding that would be most effective in helping users to create interactions. Below, we examine evidence for these aims, structured according to the categories described in Table 7.1.

7.6.1. VR authoring expertise

Since participants only analysed existing designs, we can only discuss VR authoring expertise in this regard. Our results show support for the usefulness of both VRBridge and the design aids. All participants found the Triggerset formalism easy to understand. They understood how the trigger-condition-action triads worked: "*I learned to work with the Triggersets quickly. The more you look at it, the easier it becomes and the more you get used to it."* (B1) This observation is supported by the Sequencing analysis results: all participants except one understood how the Triggersets fit together enough to identify some of their sequences. While they did not perform as well in the Debugging, participants did find some errors. The Triggersets encapsulate programming logic, especially *if…then*. This shows that end-users can understand programming constructs using the Triggersets, and how basic components can be built up into more complex interactions.

Participants from the Basic group had more trouble ordering the Triggersets in sequences than the Representation group. Only Basic group participants mentioned problems with thinking linearly. E.g., "*When*

you read something, you say OK so he goes there but you don't think that there might be something stopping movement…[it is] very hard to think about how Triggersets all fit together…[I] got more used to realising that anything can happen at any moment, but the order of things was still hard to realise." (B2) This shows that the design aids support non-linear thinking. The Representation group also considered both the spatial and timed aspects of interactions more successfully than the Basic group, as shown by their success at Sequencing. All participants had some problems understanding sequences when there were implicit player actions involved. The Representation group did better at this, which shows that the design aids support awareness of player freedom. As stated earlier, participants only had 20 minutes to examine each set of Triggersets, and were unfamiliar with the domain. Since player freedom is one of the more difficult aspects of VR interaction, we are satisfied with these results.

Participants noticed meaningful interaction details, as shown by their success at identifying sequences. However, they often did not drill down into Triggersets to find details that would allow them to be more accurate, or notice errors. This is shown most clearly in the Debugging results; e.g., most participants failed to notice the lack of a Triggerset to open the InnerDoor. The Representation group showed more expertise at noting correct details and ignoring unnecessary ones in both tasks. This consideration and usage of task-relevant detail rather than generalised problem solving is an attribute of expert performance and shows the success of the design aids in this regard.

7.6.2. Constructivist design principles

Our results show that VRBridge and the Triggersets are simple to use and understand. In addition, all participants from the Representation group stated that they found the representations generally clear and useful. Their results in both tasks support this. There were some problems with understanding the representations, e.g. two participants were unsure about what it meant when arrows in the Sequence Diagram returned to the same state (R2 and R5). This kind of knowledge support should be included in the representations.

The success of the Representation group shows that the provision of multiple representations of the interaction design led to more successful analysis. All participants explored VRBridge to build sequences and debug the Triggersets. The Representation group all looked at multiple representations, and most participants cross-referenced the representations constantly to work out what was happening in the VE. "*I had all three open at the same time. Then if you don't understand the sequence [you] can look at floorplan.*" (R2) Where participants did not explore actively was when they had to drill down into Triggersets or Timelines to find specific details, e.g. the amount of movement. This could be partly because they did not have much time to perform the tasks. However, VRBridge and the design aids can be more linked to assist in more effective exploration.

Participants identified sequences and found some errors. This shows that they were able to work alone at these fairly complex tasks relatively successfully. There was some frustration among participants in the Basic group, especially in the more complex task in Part 2. "*For Part 2, in the middle [it] got overwhelming with all of the interactions - couldn't really see what happened past that*." (B1) However, none of the Representation group expressed any frustration.

Participants from both groups engaged in reflection on their activities, e.g., "*Began still thinking linearly - didn't occur [to me] that the bell moved the bouncer and [the] bell ringing becomes its own trigger.*" (B1) and "*It reminded me of Neverwinter Nights – [it] gives a top view of world and areas that you can use to trigger things. The visualisations are useful for different things.*" (R4) Therefore, participants were able to understand the significance of the Triggersets for creating a 3D world, and its difficulties. The Representation group engaged in more reflection as they also considered the significance of each representation (this is discussed below). These results show that VRBridge and the design aids successfully embody our constructivist ideals.

7.6.3. System and interface design

At a general level, we have shown that VRBridge and the representations are usable and effective for novices. Participants discussed both in a reflective way that showed they understood the mapping between Triggersets, object details and representations; and how these mapped onto the domain of 3D interactions.

There were some problems with system visibility. These related to the difficulty of drilling down into action details and Timelines, or difficulty viewing all of the Triggersets at the same time. Therefore, participants did not find details and underused the Timelines. E.g., "*It would be useful to give more details so you don't have to open actions. It was cumbersome to open them*" (R5) and "*It is difficult to go back to an object to find its Timeline, though – [it] slows you down. That is why I did not use them much. The Timeline had too much of an extra step to work with.*" (R3). There were also times when participants did not examine representations which would have helped them: "*I didn't look at the Floorplan – that that would have told me I couldn't reach the Push-Button.*" (R6) With more help linking Triggersets to the representations, the ability to play the VE in a 3D Window and scaffolding to suggest debugging actions, these problems would be lessened.

Participants worked in different ways with the representations and Triggersets. This shows the flexibility of VRBridge in allowing designers to have different work processes. However, there were similarities in working. A fairly typical process was to begin with the Floorplan to gain an overview of the VE and its interactions. Participants would then examine the Triggersets and object descriptions, referencing names and positions of objects and locations on the Floorplan. During this process they would repeatedly examine the Sequence Diagram to find out how Triggersets sequenced, or confirm their own analyses. When Timelines

were referenced, they would examine them and check the objects and actions involved. In this way, most participants used the linking between the representations effectively.

The Triggersets and representations provided some constraints on correct actions, as shown by the ability of novices to analyse sequences, debug errors and understand requirements of VR authoring. The design aids provided valuable feedback: "*The Sequence was useful to check up after your own analysis of the triggers*." (R4) and "*It was very useful to have visualisations. Also being able to click on them, see details and work out the problem*" (R2)

More support must be provided for error handling, as shown by our results for the Debugging task. However, even in the short time provided and with unfamiliar interactions, participants did find some errors. The representations helped a lot here. No participants mentioned problems with the terminology; most had trouble understanding the results of rotations and movements, but this was about understanding 3D space.

7.6.4. Effective external representations

The discussion above shows that our representations were effective and provided benefits to participants. Representation group participants were able to work much more effectively on both sequencing and debugging, suggesting that the representations provided valuable assistance. In terms of distributed cognition, cognitive operations were assisted through the constraints and structure provided by the representations. Participants analysed Triggersets more expertly and used *lookahead* to work out their effects. They also had the benefits of cognitive offloading: they perceived important details and ignored irrelevant ones, and recognised the patterns of interaction created by Triggerset combinations. They showed increased working memory, as they created fewer external notes than Basic participants.

All but one of the Basic group stated that representations would have made their task much easier. Three specifically mentioned flowcharts and floorplans. Every Basic group participant attempted to represent the Triggersets diagrammatically, compared to only one of the Representation group, showing that the Representation group did not need to reformulate the Triggersets. Participants tried to work out the sequence with diagrams of the flow from one Triggerset to the next, often organised by Triggerset location. However, these were usually inaccurate and led to incorrect conclusions. This highlights the fact that many people do not have the skills to distil accurate overviews from one representation and diagram them in another.

The entire Representation group found the representations clear and useful, e.g., "*The Triggersets are basically just the details, just the details of the rest [the representations]*" (R6). They all felt that the tasks would have been much harder without the representations and that the interactivity provided in each diagram was very helpful.

Participants found the Floorplan most useful; it helped to orient, give a concrete sense of the space and relative object positions. E.g., "*Once you coordinate between the physical locations, you can see how you need to move.*" (R2) and "*I used the Floorplan to make the environment concrete. It was very important; I really needed it a lot*." (R5) Half the Representation group stated that they could not have reconstructed the sequences without the Floorplan. Participants used the Floorplan to view the perceptual opportunities of the VE and their significance: "*To understand the scene, the map tells you who is in it [and] where*." (R6) and "*You can understand the hierarchy once you know the relation of objects to each other.*" (R2) This helped in reconstructing the Triggerset sequences. The Floorplan therefore provided valuable spatial constraints and improved understanding of the spatial aspects of the Triggersets.

The Sequence Diagram was useful for understanding how the Triggersets and player interactions connected to the narrative: "*The Sequence Diagram is useful for seeing how the Triggersets relate, their order and what activates what.*" (R3) and "*The Sequence tells you the conditions and everything that is possible. You could walk anywhere but the VE only reacts like the Sequence*." (R6) Even those who found the Sequence Diagram less useful still found benefits: "*I used the Sequence Diagram only for a reference. The Sequence was useful to check up after your own analysis of the triggers.*" (R4). Errors were made more obvious by the Sequence Diagram: "*Mistakes were most obvious from the Sequence Diagram – that many sequences never open – [you] can't open the door.*" (R3) It also helped participants to think non-linearly about how the Triggersets would connect: "*I used the Sequence a bit, to get an idea of branches*" (R5) Participants wanted more interactivity from the Sequence Diagram. However, as they were unable to create Triggersets, they could not experience the feedback provided by the Sequence Diagram as the designer changes Triggersets.

Timelines were useful for understanding a predictable sequence "*I used the Timeline for Bouncer Move to see how he went away. The Timeline is useful to see what happens in the action – how things sequenced together*." (R3) and "*Timelines were most useful for the sequence - explains the relations between the actions of all the objects.*" (R1) In the Sequencing task, the Representations participants consistently performed better than the Basic participants on Triggersets that were replaced by Timelines (BE and WE in Table 7.7). We can therefore infer that they helped with understanding how these interactions worked together in time.

7.7. Implications for Design

We have shown the success of our interaction authoring interface: Triggersets. We have also confirmed that VRBridge is highly usable and embodies our constructivist principles. Our design aid of multiple representations has been shown to provide high-level cognitive support for more expert performance.
We also found some areas where we could improve the VRBridge interface. Some parts of the system had to be more visible and accessible, and more linked together. The ways of accessing the representations had to

be highly visible, simpler and connected to the Triggersets. We added the ability to access each representation from the others through context sensitive menus, for easier cross-referencing. We added more options to our existing context menus, creating more links between the representations and Triggersets. This also adds more interactivity for the designer to work with representations. We added the following options:

- For each object in a Triggerset, the designer can view it on the Floorplan or Object Panel (where it is highlighted). The designer can also show all objects on the Floorplan, in which case all objects involved in the trigger, condition or action will be highlighted.
- For each waypoint, location or proximity circle in a Triggerset, the designer can view it on the Floorplan where it will be highlighted. Waypoints and locations can also be viewed by selection from a menu on the InterPlay screen. Their coordinates will be shown and they will be highlighted on the Floorplan.
- If a Triggerset contains a Timeline, the designer can open it. Timelines can also be opened by selection from a menu on the InterPlay screen.
- If a Triggerset contains sounds or animations, the designer can preview them.
- We added the same options to the context menus for arrows and states in the Sequence Diagram. In addition, the Triggerset that corresponds to an arrow can be highlighted on the InterPlay screen; and Triggersets with conditions corresponding to those of a state can be highlighted in the same way.

Participants had some difficulties forming an overview of the Triggerset interface; therefore we introduced the ability to collapse the Triggersets and sort them according to description. Participants did not often drill down into Triggersets and object details to find out necessary information, e.g., the angle of rotation. As a result, we decided that details must be made more obvious. For the Representation group, the diagrams indicate many of these. We added more details to the Triggerset and Timeline descriptions.

We also found areas where scaffolding support was desirable. People have difficulty understanding the consequences of rotations and translations, especially the position of objects after a sequence. More support is needed for understanding these areas of 3D programming (3D transforms). This provided us with the major focus for our scaffolding. Other areas where scaffolding would be beneficial include:

- Understanding the significance of the Sequence Diagram. Although participants benefited from the Sequence Diagram, there were some parts which they did not understand.
- Explaining the coordinate system used in VRBridge. Some participants thought that it would work in a different way (i.e., z direction reversed, or z and y flipped), and this confused them.
- Debugging interactions, especially highlighting inconsistent effects. The ability to run the VE in a 3D Window should help with this, but support can be provided by indicating typical errors.
- General support for using VRBridge, so that the system status and authoring actions are more visible. This will help designers understand how to begin authoring.

- General support for using multiple representations effectively, especially highlighting how they can assist in design, the domain knowledge embedded in them and how they connect to the Triggersets.

7.8. Synthesis

We have described our experiences in providing multiple representations to support understanding of VE interaction design. In this study, the Representation group consistently out-performed the Basic group. They understood the representations and worked well with them, naturally using multiple interacting representations to build a more complete idea of the design. Floorplans were used to gain an overview of the VE and interactions, Timelines were used to understand the consequences of predictable sequences of events, and Sequence Diagrams were used to understand how the interactions worked together and to find errors. These results show that using multiple representations to decompose, understand and debug VE interactions improves the authoring process. This provides us with validation of the efficacy of our representations, which creates a solid basis for the next design iteration.

We also evaluated our VRBridge interfaces and the Triggerset mechanism. We found that all participants, even those who did not receive representations, were positive about the Triggersets and found them easy to understand for programming interactions.

We found some areas where we could improve existing VRBridge interfaces with the addition of context menus for stronger linking between Triggersets and representations; and more visible descriptions for actions in Triggersets and Timelines. We also found areas where the provision of scaffolding would be most useful.

These results show that using multiple representations to understand and debug VE interactions substantially enhances the authoring process and supports increased expert-like performance in this domain. This confirmed our design decisions for VRBridge. In the next chapter we describe the addition of the 3D Window, dynamic linking and scaffolding to VRBridge.

8. VRBridge: Design of 3D Window and Scaffolding

This chapter describes the second design phase of VRBridge. We begin by briefly reviewing the results of the previous design cycle; then we describe the additions made to the system: the 3D Window, Interaction Engine and external engines; dynamic linking between the representations, Triggersets and 3D Window; and scaffolding.

8.1. Review of Previous Design Cycle

In Chapter 6 we described the design of VRBridge, focussing on the interaction programming interface for Triggersets and the design aid of additional representations. We also discussed how the design of VRBridge and its design aids reflected our constructivist design principles; and was informed by our research on expert and novice practices. We designed the Triggersets, Floorplan, Sequence Diagram and Timelines to be easy to use for novices, and powerful enough to express the interactions for a variety of simple VEs.

In Chapter 7, we described the evaluation of our design without a 3D representation; this was done so that we could test the expressiveness of the Triggersets and representations. We also evaluated the ease of use of VRBridge and determined the areas where scaffolding would be most useful. Our evaluation showed that all participants were able to understand the Triggersets to some extent; and those participants who had access to the additional representations performed better than those who did not.

8.2. Adding the 3D Window, Interaction Engine and External Engines

To frame our description of the 3D Window and engines, we redisplay the graphical representation of our architecture in Figure 8.1. We described the design of the Mediator, InterPlay screen and representations in Chapter 6; in this section we build on our initial design and complete the description of our architecture by discussing the design and addition of the 3D Window, Interaction Engine, render engine and sound engine. The Interaction Engine and 3D Window communicate through the Mediator and directly with each other. The external engines, which are plugged in to the system, only communicate with the rest of VRBridge through the 3D Window as all communication must be translated into generic functions for 3D interaction.

8.2.1. The 3D Window class

The 3D Window is central to any VR system as it displays the world with which the player interacts. The 3D Window is the designer's window on the 3D world. At any time in the design process, the designer can open the 3D Window and view the effects of Triggersets. It plays out the interactions described by Triggersets and allows the designer to experience them personally; therefore it makes the abstract interactions more concrete.

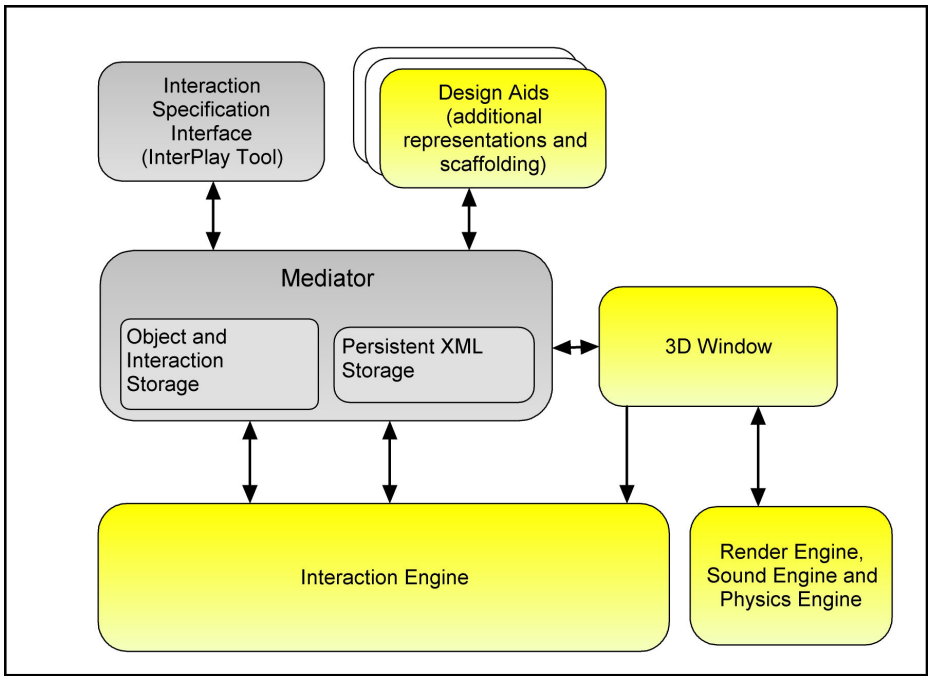


Figure 8.1: VR Bridge architecture, showing the main modules and how they interact. The highlighted modules are described in this chapter, while the greyed out ones were described in Chapter 6.

The 3D Window module manages the VE and its interface. It connects output from the rest of VRBridge to the displayed world. Behind the 3D Window interface is a module which translates generic functions for 3D interaction into commands for specific render and sound engines, and vice versa. This modularity allows the external engines to be replaced by simply modifying the translation module. The generic functions correspond to actions for displaying a dynamic 3D world, including actions described by Triggersets (e.g., translating an object), object properties (e.g., collision detection), starting the world and 3D Window (e.g., creating devices for rendering and sound), and creating the objects and geometry of the world (e.g., instantiating and placing object models). The module also contains functions for the designer to manipulate the 3D world while authoring, e.g., pausing the action, and switching between first-person and bird's-eye views.

We used the open source render engine, Irrlicht[3], as it was freely available and powerful enough for our needs. For sound, we used the sound engine, IrrKlang[4]. We chose not to include a physics engine as Irrlicht provides a basic physics simulation. However, one could easily be added by modifying the translation module. All objects and sounds included in the VRBridge library were freely available on the internet.

[3] http://irrlicht.sourceforge.net/

[4] http://www.ambiera.com/irrklang/

A Preview Window class inherited from the 3D Window is used for previews connected to the Object Panel. These allow designers to view object models and animations, and listen to their associated sounds. Thus, the designer gains feedback on the length of animations and sounds without instantiating them in the 3D world. This provides assistance for considering 3D world content easily and efficiently during the design process.

8.2.2. The 3D Window as output window

The 3D Window allows the designer to experience the VE using various views. The most important is as a player by showing a first-person view through the player camera. Buttons on the 3D Window allow the designer to switch between the first-person view (or *Walk Mode*) and *Fly Mode*. With *Walk Mode*, the camera is subject to gravity and collision detection. This means that the designer can be part of the VE action like any other object. With *Fly Mode*, both gravity and collision detection are turned off. This means that the designer can view the VE from different angles and positions; if she 'flies up' to view the VE from above, the 3D Window corresponds directly to the Floorplan. Designers can use this to orient themselves, gain an overview of the VE, and make the link between the 3D world and the Floorplan concrete.

In addition to *Walk/Fly Mode*, the designer can *Pause* the VE and toggle the *Run Mode* using buttons. Run Mode provides dynamic linking between the 3D world and representations, and is described in Section 8.3. The ability to pause the VE action and examine its current state (the player camera can still move) is useful for fine-tuning interactions and understanding errors. The buttons clearly show their state to avoid confusion.

Research shows that, for effective 3D perception, providing a clear reference ground plane and placing recognisable objects on it helps with self-orientation (Ware 2000). Because of this, the 3D Window opens with recognisable objects already placed (e.g., furniture), and the limits of the world are clearly defined with walls and floor. This provides constraints for designers, and also supports creativity as the objects in the world may inspire designers to explore interaction possibilities. Figure 8.2 displays a first person perspective on the 3D world and Figure 8.3 displays a bird's eye view.

Waypoints and locations are not visible in the 3D world as they are not part of it; they are constructs used by the designer to facilitate movement and rotation in the world. However, they can be made visible by selecting menu items at the top of the 3D Window. Waypoints will appear as shiny gold pillars and locations will appear as shiny gold rectangles on the ground. This creates another connection between the 3D world and the other representations, as the designer can view waypoints and locations on the Floorplan and in the 3D world, and link to them to the Triggersets, Sequence Diagram and Timelines. This provides the designer with a strong spatial idea of where various interactions happen and how objects must move across the space to trigger them. The 3D Window can be juxtaposed with the other representations for direct comparison. This helps designers to disentangle their interactions, as they compare the various representations.

Figure 8.2: 3D Window in Walk Mode or first-person view, showing two active objects. The upright rectangle on the right is a waypoint-marker.

Figure 8.3: 3D Window in Fly Mode, showing the environment framed by a sky box. The rectangle on the left is a location-marker and the upright rectangle between the plants is the waypoint-marker seen in Figure 8.2.

8.2.3. The Interaction Engine

Just as the 3D Window translates between generic 3D functions and render engine specific functions, the Interaction Engine translates the Triggersets into 3D interactions when the VE is running. It contains functions which calculate when and which interactions will occur under the specified conditions in each frame of an executing VR. It passes information about the effects of interaction to the Mediator, which makes the required changes to the content, manages the frame loop, and passes the changes to the 3D Window and the appropriate representations. A graphical depiction of this process is displayed in Figure 8.4.

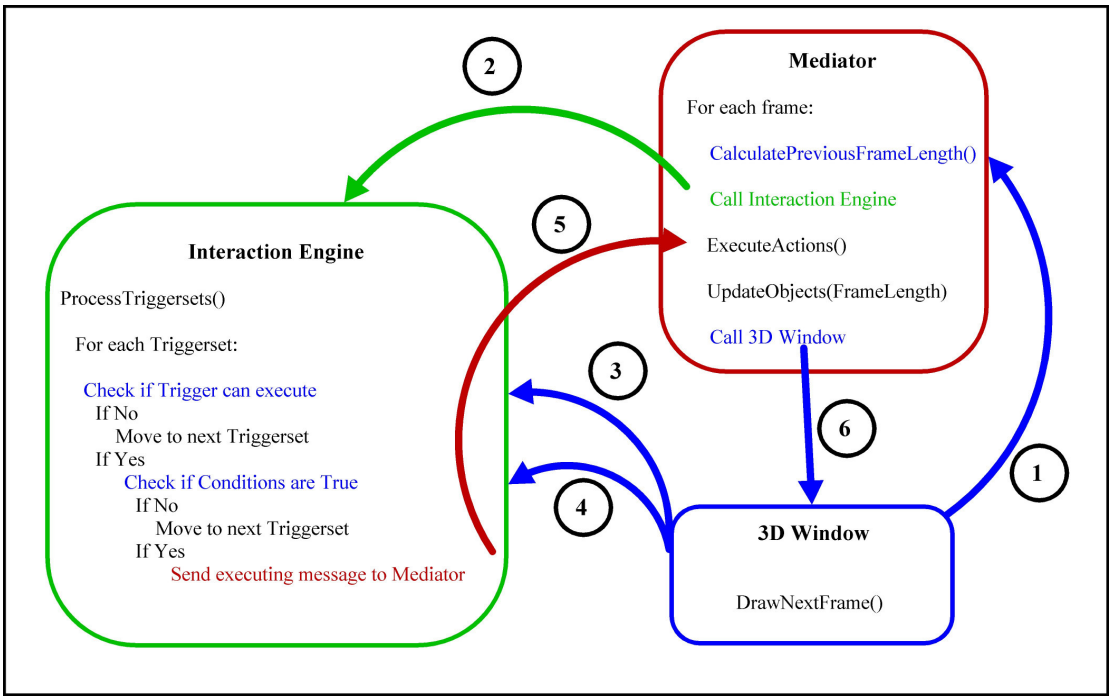


Figure 8.4: Graphical depiction of the process for calculating interactions in the 3D world, showing the flow of information between the Mediator, Interaction Engine and the 3D Window.

The numbered arrows show the flow of information and control during the execution process. The Mediator controls the frame loop.

- For each frame, the Mediator requests information from the 3D Window about the length of the last VE frame length (1). This length indicates how much time has passed since the last request, which provides information on the interactions (or parts thereof) that occurred during the frame.
- Then the Mediator calls the Interaction Engine, which contains the Triggersets and functions for working with them (2). The Interaction Engine's core function is ProcessTriggersets(). This examines each Triggerset and works out whether it can execute.

- First, it checks the trigger and requests information from the 3D Window (3), e.g., whether the player is pressing a key, or positions of key objects. Using this information it works out whether the trigger can execute, e.g., it calculates whether two objects in a proximity trigger have just entered the specified distance from each other. If not, then the current Triggerset cannot execute and the next one is checked.
- If the trigger can execute, the Triggerset's conditions are checked, also with information from the 3D Window (4). The process for calculating whether conditions are true is the same as that for trigger execution, except that an instantaneous event is not required, e.g., the function returns whether two objects are in proximity, and does not include information about whether this has just occurred. If any of the conditions are not true, then the current Triggerset cannot execute and the next one is checked.
- If all of the Triggerset conditions are true, then the Interaction Engine sends a message to the Mediator, indicating which Triggerset is executing (5). The Mediator executes the actions of the Triggerset by updating variables connected with each object, which indicate whether it is performing an action. E.g., an active object keeps a variable for the current animation, its length, speed, and whether it repeats.
- When the actions of executing Triggersets have been updated, the Mediator calls UpdateObjects(). This function cycles through all VE objects and calculates their changes based on the actions. This includes Triggersets that have executed in previous frames and whose actions are still active. E.g., if a move action was triggered for an object, its position will be updated on each frame by using its speed and the frame length, until the action is finished. We handle conflicts through interruption as this is a simple mechanism which should be easier for novice users to debug when unexpected conflicts occur.
- Finally, the Mediator calls the 3D Window (6), which draws the scene using the render engine.

8.3. Adding dynamic linking between the representations

Having added the 3D Window to VRBridge, we added dynamic linking. In creating dynamic links, we leveraged people's perceptual abilities by using pre-attentive processing as much as possible. As described in Section 3.1.1, there are four categories of elements which pop out in this way: form, colour, spatial position and motion. In Chapter 6, we described how we used most of these in our design. However, we had not yet added motion, except in the use of flashing highlights. As mentioned previously, the use of motion must be carefully planned as too many moving elements can distract attention rather than focussing it.

We have already added some dynamic linking with our context menus, described in Section 7.7. These provide increased coupling of views, as the designer can select elements from the context menus on Triggersets and each representation (e.g., a timeline that is referenced), and view their depiction elsewhere. E.g., the designer can view the position of a waypoint used in a Triggerset on the Floorplan. In this section we describe additional dynamic linking through slaved views and the provision of a configurable Run Mode.

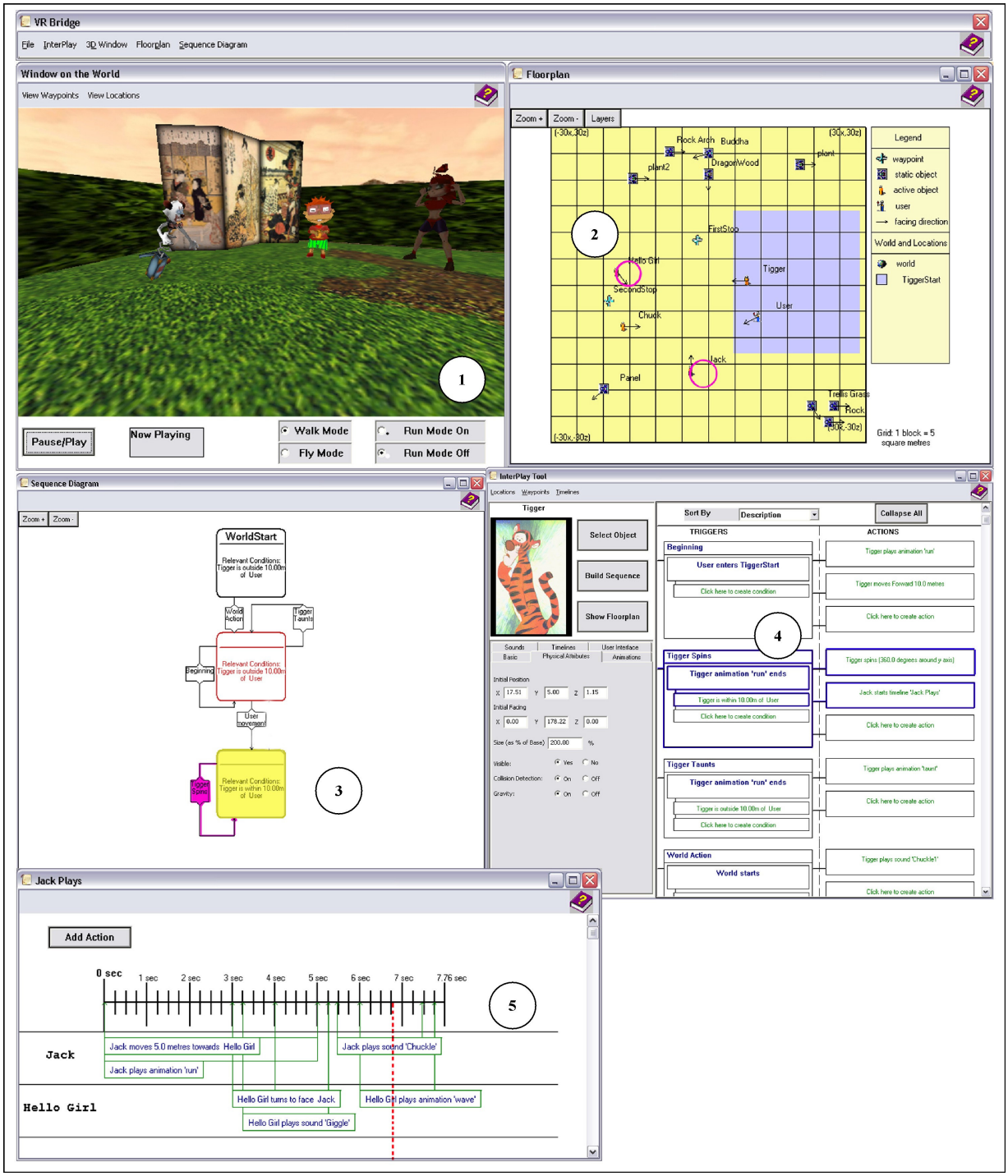


Figure 8.5: Screenshots of the 3D Window, the InterPlay screen, the Floorplan, the Sequence Diagram and a Timeline, with numbers showing where each view is slaved to the 3D Window while it is running.

We added linking between the additional representations and the 3D Window. When the 3D Window is open and the VR is running, all other open representations are slaved to it. Therefore the interactions and changes

in the 3D world are reflected in the rest of the system. Figure 8.5 shows how the InterPlay screen and additional representations are slaved to the 3D Window when it is running. In the 3D Window (1), the player has moved towards the Tigger object and triggered a sequence of interactions, where Tigger runs forwards and triggers a Timeline in which the objects, *Jack* and *Hello Girl* perform actions. On the Floorplan (2), the position of the *player* has changed and the objects that are currently moving are highlighted (*Jack* and *Hello Girl*). On the Sequence Diagram (3), the state of the world in which the *player* and *Tigger* are close to each other is highlighted, as is the arrow that is currently executing. This corresponds to a Triggerset, *Tigger Spins*, which is also highlighted on the InterPlay screen (4). On the Timeline (5), a moving highlight shows how the action is progressing.

The Run Mode uses the slaved views with the running 3D world described above. It synchronises the representations, the 3D Window and the InterPlay screen so that they are open and developments in the VE can be viewed in all of them. Automatically opening the representations scaffolds the use of individual dynamic links by novice designers, as they can see the usefulness of having connected multiple views. However, we were concerned that increasing numbers and varieties of distracters might reduce the effect of highlighting, especially in a limited amount of screen space. We also did not want to take away the control of the designer by launching multiple views every time the 3D world was started. Therefore, we made the Run Mode an option that could be toggled by the designer. Figures 2 and 3 show the buttons for controlling the Run Mode. The windows are smaller than their usual default sizes (all windows can be resized in VRBridge), which creates a more iconic view. This is not useful for detail but should clarify patterns in the highlights in each view and across views. The position and size of any of the windows can be changed and any window can be closed once the Run Mode has opened, allowing it to be configured.

Once the 3D world is running, the representations and InterPlay screen are slaved to the 3D Window as described previously. As the designer moves through the VE, a highlight moves through all of the visualisations, indicating where the action is taking place. If the 3D Window is paused, the highlights in the visualisations are also paused. This should encourage reflection as the representations can all be viewed simultaneously at a pace chosen by the designer. It powerfully connects the different views with the 3D world, and helps the designer to understand how they relate to each other. It is also a useful debugging technique, as the cause-and-effect relationships between Triggersets and actions in the 3D world are visualised in multiple ways, which should make errors more apparent.

8.4. Adding scaffolding

Our scaffolding research (described in Section 3.2) showed that many definitions would include our representations, locations and waypoints, and general system design as scaffolding. The definitions agree

that it is support which helps novices to accomplish tasks that would otherwise be beyond their ability. However, theorists disagree about whether support which is ongoing and does not fade as the novice progresses should be included as scaffolding. As discussed in Section 3.2.1, we believe that it is useful to consider both conceptions of scaffolding, while making a distinction between them. In this section, we focus on the more strict definition of scaffolding, and describe the design and addition of multiple scaffolds to VRBridge. However, first we consider how various parts of the existing system support end-users, in terms of the initial benefits provided by scaffolding to learners.

8.4.1. Considering VRBridge as support

VRBridge tries to support a different mode of thinking in the novice, which approaches how the VR interaction expert thinks. Describing VRBridge in scaffolding terms helps us to explicitly consider how it provides support. Wood et al. (1976) initially defined scaffolding in terms of six functions, which we can use to describe the support provided by VRBridge. During this analysis, we reference the Scaffolding Design Framework of Quintana et al. (2002), which we described in Section 3.2.3, and which overlaps with Wood et al.'s functions. To recap, the Scaffolding Design Framework considers three broad cognitive challenges of learners and guidelines for addressing them: process management, which can be addressed through structuring tasks and functionality and providing access to expert knowledge; sense making, which can be addressed through using representations that can be explored by learners and bridge learner and expert knowledge; and articulation, which can be addressed through helping learners to describe and explain their work.

To engage interest in the task – VRBridge makes adding dynamic interaction to the 3D world enticing by providing attractive content which the end-user can explore. The interfaces are designed according to user interface principles to be attractive and inviting.

To maintain interest in completing the task – By making each step simple, exploration and recovery from errors is easy. The act of creating more complex interactions as the user learns should maintain interest. This function and the one above do not fit into the Scaffolding Design Framework, as it does not consider how to engage and maintain interest. However, support is only useful if the learner engages with it.

To constrain and simplify the task – This function is part of *process management*. VRBridge decomposes interaction authoring so that the designer can focus on different parts of it, e.g., usage of space and time. VRBridge removes high-level programming through the use of Triggersets, which simplifies the process of creating interactions. The Floorplan supports understanding of space in the VE by providing a simple 2D representation; the locations and waypoints help the designer to accomplish spatial interactions more easily

and effectively. While these could be scaffolds, even expert VR designers use maps and waypoints for understanding 3D space, showing that their usage is unlikely to fade (Ware 2000, Darken et al. 2002).

To highlight critical features of the task and discrepancies between the learner's actions and the task requirements – This function is part of both *sense making* and *articulation*, since when critical features are highlighted they help the user to make sense of what is happening in the task, and to explain it. Our representations highlight important or problematic parts of VR interaction authoring, and guide the user in considering them effectively. They help users in debugging interactions, an activity that is principally concerned with highlighting discrepancies between actions that have been taken and what should have happened. The Sequence Diagram in particular highlights how the interactions connect and encourages reflection on the authoring task. Although it can be complex to examine, and adds effort to the task rather than making it easier, this acts to uncover important aspects of Triggersets. In the same way, the Run Mode highlights how the various representations connect with the 3D world and its interactions.

To control learner frustration – This function is part of *process management*, since if the task is structured carefully, user frustration will be lessened. VRBridge allows errors to be recovered from easily, as creating interactions consists of simple steps. This should decrease user frustration. The Floorplan also controls frustration by providing an easily understandable overview of the 3D world. In a similar way, Timelines make synchronising interactions easier, as they can explicitly be organised together without constant flipping between the 3D world and the programming.

To demonstrate correct actions to take in completing the task – This function is part of *sense making* and *process management*. By structuring tool functionality, we guide the user towards performing correct actions; and by embedding expert knowledge in representations and other parts of the system, we support the user in understanding how the expert works and thinks about the task. As can be seen from the above, different levels of help are provided by VRBridge. Our external representations have elements of scaffolding; however they add support for the VR interaction authoring task which an expert would use. Specifically, the Floorplan adds a spatial overview and a way to place markers and areas in space easily; the Timelines make synchronising actions easier and less time consuming; and the Sequence Diagram helps users reflect on interactions and debug them.

8.4.2. Scaffolding support

Below we describe the VRBridge scaffolding. We designed it with principles derived from our constructivist theory and the research on scaffolding. Our requirements were: multiple types of scaffolding to address different issues; non-invasive scaffolding under user control, including the ability to make it fade; and

scaffolding that encourages users to explore the domain of dynamic VR and to reflect on the meaning of their authoring actions within the wider context.

As described in Section 3.2.1, Jackson et al. (1998) defined three types of scaffolding which we can use to provide complementary support:

- Supportive scaffolding provides support and advice for performing tasks without changing them.
- Reflective scaffolding provides support for thinking about tasks without changing them.
- Intrinsic scaffolding changes tasks, e.g., by reducing their complexity or focussing attention.

The problems addressed by the scaffolding are those identified during our user studies. To recap, we found that users need more support in understanding parts of 3D programming e.g., 3D transforms; effectively working with representations, especially the Sequence Diagram; clarification on concepts such as the coordinate system used in VRBridge; using the VRBridge interfaces; and debugging interactions. To address these problems, we devised three forms of scaffolding which we added to VRBridge:

- *Tooltips* explain terminology and how to work with the system and the representations;
- *Context Sensitive Hints* encode domain knowledge, explain concepts, and provide debugging guidance;
- *Wizards* for addressing more difficult parts of 3D interaction design, such as 3D transforms.

8.4.3. Tooltips

Tooltips are supportive scaffolding, as they do not change the task and they provide advice about how to do certain aspects of it. We designed over fifty Tooltips to provide information and advice about performing actions. Our main focus was to provide brief, constantly available guidelines about how to use VRBridge, and actions to take within the domain of VR interaction. We designed Tooltips to appear after a hover time of 250 milliseconds and remain until the mouse was moved. Therefore, they appear quickly enough to be associated with current actions (Ware 2000). Table 8.1 describes the information provided by the Tooltips.

Focus Area	**Specific Purpose**	**Examples**
VRBridge and Interface	Explain interface functionality, e.g., buttons and dialogs.	*The tabs and picture on this panel describe the selected object. You can change object details here.*
	Describe how to perform important actions with VRBridge.	*An arrow* (Sequence Diagram) *describes a Triggerset or user movement. It goes from a state where the action is possible to the state which results if the action happens. Left click on an arrow to show the Triggerset.*
Domain-specific	Describe how to perform important steps in creating VR interactions.	*To move the start time of an action* (on the Timeline)*, left click on the top left corner of the action and drag to the new time.*

Focus Area	Specific Purpose	Examples
	Describe domain-specific terminology.	*Create a collision trigger by selecting the two objects which must touch.*

Table 8.1: Table of various kinds of information provided by Tooltips, with examples.

VRBridge opens with the Tooltips turned on. However, they can be toggled from the main menu. In this way, the user can control whether they appear, and they will fade when turned off. Tooltips are the only form of additional scaffolding that appears automatically, rather than being requested by the user. Figure 8.6 shows a Tooltip on the Floorplan in VRBridge.

8.4.4. Context Sensitive Hints

Context Sensitive Hints represent both supportive and reflective scaffolding: they do not change the user's task; they provide advice about performing actions; and they provide support for thinking about the task in domain terms. We use them as well as Tooltips to describe certain aspects of the system and domain for two reasons. Firstly, research showed that learners often ignore help unless they open it themselves, which is not the case with the Tooltips. Secondly, certain information requires more description than is appropriate with Tooltips, which must be kept short because of their frequency and ubiquity.

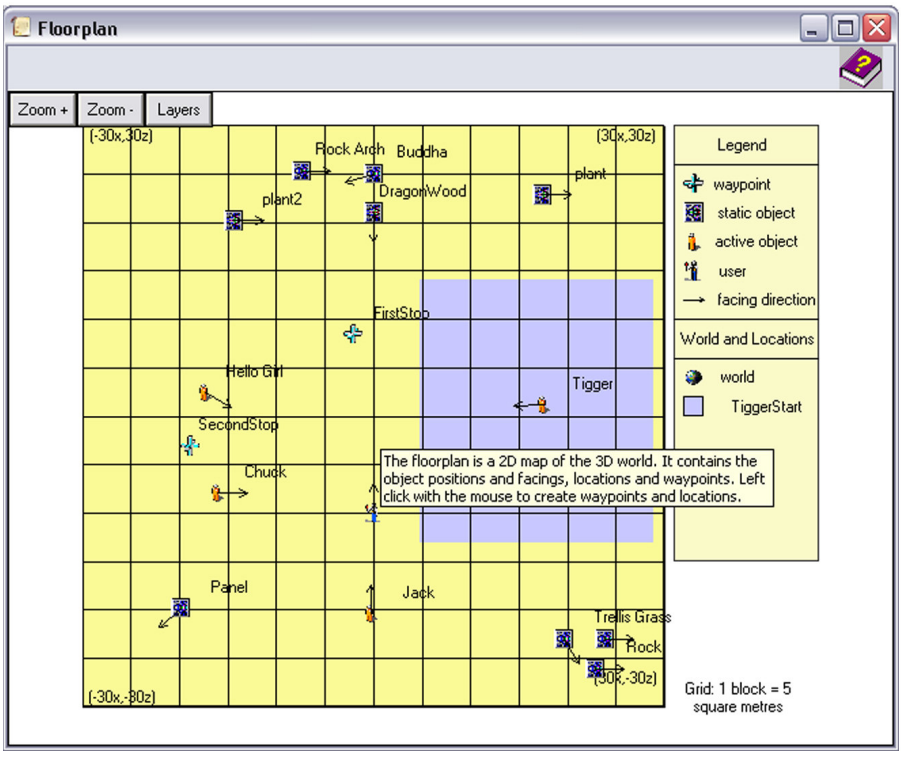


Figure 8.6: Floorplan of VRBridge, showing tooltip and icon for Context Sensitive Hints.

Context Sensitive Hints are accessed by selecting the help icon in the right corner of the menu-bar on each VRBridge screen. This is consistent across screens for easy recognition. Figure 8.6 shows the icon on the Floorplan. Each icon opens a menu of Hints specific to that screen. When this is selected, a window appears with the Hint. We include examples in each Hint, to provide both an abstract and concrete description.

We use Context Sensitive Hints to provide design and debugging advice, and to provide domain specific information (e.g., about 3D world coordinate systems); we chose information which would be useful to end-users in gaining a deeper understanding of how to design VR interactions effectively; and understand cognitive strategies of experts in the domain. In this way, we provide domain knowledge, and support reflection about design within the VR domain and metacognition about the user's own design process. Table 8.2 describes each context sensitive hint, organised by screen.

Screen	**Context Sensitive Hint**	**Description**
Main	*How to start*	Help system, how to open each screen and what it does.
	Saving and loading your design	How to save and load XML files. How XML files hold all VE information.
	3D coordinates	3D coordinates as the dimensions of a box, and related to the real world. How objects are placed at a 3D point.
	How VR works	How to design from the player perspective in VEs and the importance of designing non-linear interactions.
	Turning tooltips on and off	How to toggle tooltips.
InterPlay	*How to create actions in the 3D world using Triggersets*	How to create a Triggerset and view it in the 3D world.
	Making sure your Triggersets connect to each other	How to make Triggersets operate effectively, e.g., how they can be activated, creating an action to trigger things, and creating a variety of Triggersets.
	The object panel	The object panel and how it can be used to change object details. How it relates to the Floorplan.
	The effects of object gravity	How to toggle gravity on the Object Panel, and the effect this has on an object and the player.
Floorplan	*How to use the Floorplan*	What the Floorplan shows and how to use it. How it is synchronised with the 3D Window.
	Using waypoints and locations	Waypoints and locations, how to create, move and delete them on the Floorplan, view them in the 3D world, and how they can be used to work with the space.
Sequence	*How to use the Sequence*	What the Sequence Diagram shows and how to use it. How it

Screen	Context Sensitive Hint	Description
	Diagram	is synchronised with the 3D Window.
	What is a state	What a state is and how it relates to the 3D world.
	What is a player arrow	What player arrows are and how they relate to Triggersets.
	What is the world start state	The world start state and how it is different to others.
	What it means when Triggersets do not appear	What it means if a Triggerset does not appear on the Sequence Diagram; also suggests debugging actions for typical mistakes.
	What it means when Triggersets return to the same state	What it means if a Triggerset returns to the same state that it left.
Timelines	*How to use Timelines*	What a Timeline is and how to use and access it. How it is synchronised with the 3D Window.
	Changing action starting times	How to change the time that an action starts.
	Changing or deleting actions on the Timeline	How to change and delete actions, including how to make them longer, or synchronise them with other actions.
3D Window	*How to use the 3D Window*	How to use the 3D Window, including how to view waypoints and locations; and use pause, walk, and fly.
	How to use the Run Mode	How to use the Run Mode, with examples of how it can help to understand how actions are happening.
	What it means if an object disappears from the 3D world	What it could mean if an object disappears and actions to find the object and fix the problem.
	What it means if an object gets stuck in the 3D world	What it could mean if an object gets stuck in the 3D world, discussing bounding boxes, and how to fix the problem.

Table 8.2: Table showing context sensitive hints in VRBridge, organised by the screen on which they appear.

As the user chooses to open the Context Sensitive Hints, they are effectively faded if they are ignored.

8.4.5. Wizards

Wizards are intrinsic scaffolding as they change the user's task by reducing its complexity and focussing attention on its critical aspects. However, they also have both supportive and reflective elements: by decomposing the task and allowing the user to step through it, with additional guidance and explanation for each step, the user is provided with advice and the opportunity to reflect on the task. We use Wizards to provide additional learning support for those 3D interaction tasks that are typically highly problematic for

users. Although VRBridge as a whole provides this support, some aspects of 3D interaction design are more difficult to understand than others. An important purpose of our scaffolding is to provide additional support for these aspects.

While the Wizards change the task of the user, they do not add functionality that is not already provided by VRBridge, i.e., they duplicate tasks that can be performed with VRBridge. What they do add is increased explanation about aspects of the tasks, various ways of accomplishing them effectively, and more immediate feedback about actions. The Wizards are the most interactive form of scaffolding that we provide, as the user works with the Wizard, actively performing suggested steps in a task. Given our principles of user control, exploration and reflection, it was important to provide Wizards which would require the user to make choices and explore the complexities of the task. Because a Wizard is separate from the system, it provides a simpler and safer environment to explore difficult tasks, without consequences for the rest of the design.

We designed two Wizards for our scaffolding. In a complete system more should be provided for users to safely explore difficult sub-tasks; however, we decided that a small number would be more useful for evaluating the effectiveness of our Wizard design and its impact on user performance. There were two areas of 3D interaction design that clearly needed additional support: moving objects in the 3D world and creating 3D rotations. Therefore, we created a Movement Wizard and a Rotation Wizard. All of the screens in both Wizards have a consistent appearance for recognition and familiarity. As both Wizards describe the task of creating specific actions, they are accessed from the action menu of each Triggerset. They output Timelines which contain the actions that have been created. The user can then choose to add the Timeline to the existing design, view it in the VE, modify it or delete it. This scaffolds expertise in creating movement and rotation actions, as the user can see how various actions combine to create a compelling interaction. Both Wizards are described briefly here, and more fully (with screenshots) in Appendix C.

The Movement Wizard describes how to create a movement interaction, with rotations to face the direction of movement. Effectively designed movement is more realistic when the object is performing animations and sounds. Therefore, we included the ability to add these to the action, to scaffold the unassisted design of realistic movements and introduce domain knowledge. The Wizard helps the user to select the moving object (with a preview), the type of movement, its distance, duration and speed. The options for movement type are:

- move towards another object or waypoint;
- move towards a position selected on the Floorplan;
- move forward, backward, up, down, left or right as currently facing (shown in Figure 8.7).

Each option provides a mini-Floorplan which guides the user visually by highlighting the objects involved, showing their coordinates and the distance to move. Duration and speed are entered into connected edit boxes. Therefore, users can enter details for one and watch how the other changes, which helps them to

understand the connection between how fast the object is moving and how long it will take to perform the action. The screen for adding animations and sounds to the movement contains preview windows for both.

The Rotation Wizard describes how to create a rotation interaction. As with movement, we included the ability to add animations and sounds to the rotation. The Wizard helps the user to select the rotating object (with a preview), the type and degrees of rotation, its duration and speed. The options for rotation type are:

- rotate to face another object or waypoint;
- rotate to face a position on the Floorplan;
- rotate by spinning, cartwheeling or flick-flacking (corresponds to rotation around y, z and x axes).

Each option provides a mini-Floorplan which guides the user visually by highlighting the objects involved, showing their coordinates and the angle to rotate. The screen for rotation by axis (displayed in Figure 8.8) provides support for understanding axis rotation, as it shows a diagram of the rotation alongside the preview window, and connects this to the amount that is being turned. It also connects revolutions and degrees, so that the user can understand how they relate. The screens for adding duration and speed, and animation and sound are similar to those for movement.

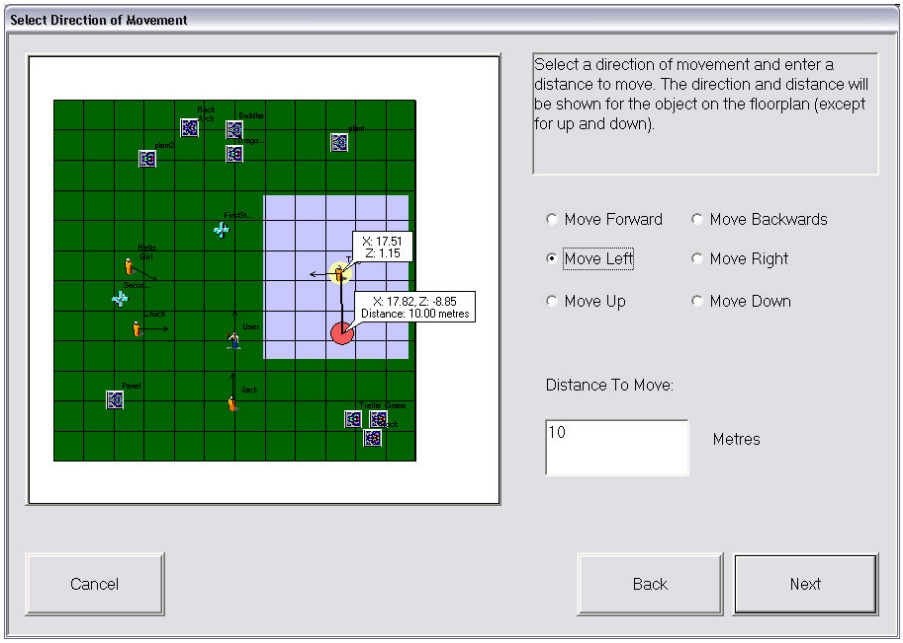


Figure 8.7: Screenshot of Movement Wizard, showing support for moving an option for moving an object in a specified direction and distance. The Floorplan appears with markers and object details.

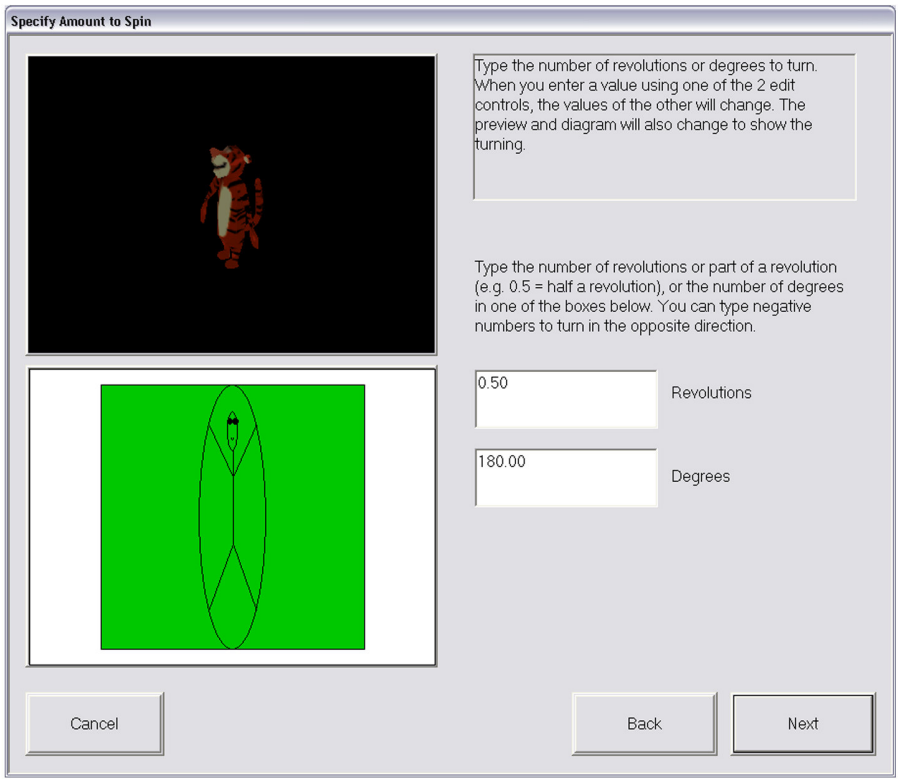


Figure 8.8: Screenshot of Rotation Wizard, showing how the user selects the amount to turn around an axis.

The Wizards provide domain level support and knowledge. While helping the user to create rotations and movements more easily, they also show the user how to do this more effectively. In this way, they highlight aspects of actions that experts in the domain would consider, and help the user in composing actions (not just creating a basic action without refinement). They also highlight the correspondence between general and domain-specific terminology, e.g., spin versus rotation around the x axis; and revolutions versus degrees. In addition, by showing the distances and degrees that are selected on the Floorplans, we encourage the user to understand how the space of the VE and actions work together. Providing Timelines for the output of each Wizard allows the synchronisation of the actions to be explicitly shown; it also allows the actions to be encapsulated and kept separate from the rest of the system, so that they can easily be deleted or edited.

Although Wizards are intrinsic scaffolding, they can easily be faded in the same way as the Context Sensitive Hints, i.e., they will no longer be used when the user stops opening them.

8.5. Synthesis

In this chapter, we have described the addition of the 3D Window, dynamic linking and scaffolding to VRBridge. The 3D Window is integral to any VR system and creates the opportunity to add dynamic linking

to the representations. Our scaffolding was designed to provide different kinds of support, with various requirements for interactivity and usage. The tooltips provide brief low-level advice, the context sensitive hints provide more detailed abstract advice about the system and domain (with concrete examples), and the wizards provide concrete support for performing particularly difficult tasks. In this way we provide various kinds of support for using VRBridge and for domain-specific understanding. Table 8.3 displays a summary of the scaffolding that we provide, categorised by type.

Type of Scaffolding	**Scaffolding Provided**	**Example of support**
Supportive	Tooltips	*Left click on an arrow to highlight all arrows of this type, or to show the Triggerset which corresponds to the arrow*
	Context Sensitive Hints	*Describes the object panel and how it can be used to change object details*
Reflective	Context Sensitive Hints	*Describes how to design from the player's perspective*
	Wizards	*Highlights the connection between axes of rotation (x, y, z) and more familiar terminology (spin, flick flack, cartwheel)*
Intrinsic	Wizards	*Simplifies the creation of rotations and movement in 3D in a step-by-step process*

Table 8.3: Table showing the three types of scaffolding, how we provided it in VRBridge, and examples of use.

Most of our decisions for scaffolding and dynamic linking were influenced by the results of our evaluation described in Chapter 7. That evaluation focussed on using VRBridge for analysing and understanding Triggersets, using the additional representations. Our next step is to evaluate VRBridge again, this time focussing on how it may be used to construct interactions in a VE effectively. This evaluation will examine how well the additional representations, scaffolding, and their combination help users to construct several typical tasks and create a VR with them.

9. Evaluation of VRBridge, Representations and Scaffolding

Having completed implementation of VRBridge, we could evaluate the full system. Our previous study (see Chapter 7) assessed Triggerset analysis supported by representations. The focus of the current evaluation was VE interaction creation using the design aids. We also wanted to evaluate VRBridge's learning curve and how design aid usage changed over time. Therefore, we designed a longitudinal experiment, comprising two authoring sessions, one week apart.

9.1. Experimental Design

The experiment was a 2x2 between-subjects and a 1x2 within-subjects repeated measures design. The between-subjects independent variables were Scaffolding and Representations; and the within-subjects independent variable was Visit (the first and second sessions with VRBridge). The design required four groups, described below and in Table 9.1:

- The Control group received the basic VRBridge system with Timelines.
- The Representation group received representations, i.e., Floorplan, Sequence Diagram and Run Mode.
- The Scaffolding group received Wizards, Context Sensitive Hints and Tooltips.
- The Combination group received both scaffolding and representations.

	Representations	**No Representations**
Scaffolding	Combination group	Scaffolding group
No Scaffolding	Representation group	Control group

Table 9.1: Experimental design for evaluating VRBridge, showing the four participant groups.

We provided Timelines to all groups for two reasons:

- They are integrated with the system, as they are used to construct interactions. Although Timeline functionality can be replicated with Triggersets, this often requires more time and creative linking. We did not want to undermine the authoring interface, as this would lessen the value of our evaluation.
- Since Timelines are provided to all participants, their output can be scored without knowledge of experimental group, as all groups can produce Timelines. This increases the evaluation validity.

9.2. Hypotheses and Aims

Our hypotheses for this experiment were as follows:

1. Both representations and scaffolding will improve participant performance, which we define as accurate and skilful task completion, effective exploration, satisfaction and learning. Representations

will improve task performance, effective exploration and satisfaction more and scaffolding will improve learning more. These distinctions are based on the success of representations in improving performance, satisfaction and exploration in our previous evaluation, and the guidance that scaffolding provides in assisting learning. The combination of design aids will improve performance the most.

2. All groups will complete the task more accurately and be more satisfied in the second session.
3. Participants will use the scaffolding less in the second session. We wanted to observe how the scaffolding was faded.
4. The Floorplan will be most used and provide most satisfaction (based on our previous study). We were interested in usage of the representations, compared to each other and across sessions.

9.3. Materials

In this section we describe the experimental tasks and materials provided.

9.3.1. Designing benchmark interactions

We designed benchmark interactions for the experiment, comprising actions that typically occur in a VE. These have been defined previously in our research on VR interactions (see Section 4.2.3), our system design (see Section 6.3.2) and our first evaluation (see Section 7.2.1). They use object properties, such as sounds, animations and movement to add dynamism to the VE. The interactions require the designer to compose actions meaningfully and to include the player. Fencott's (1999) criteria for effective VEs of agency, narrative potential, presence and transformation (see Section 4.2.2) inspired these benchmark interactions:

- *Object reactions to player interaction* – objects react to the player meaningfully and appropriately.
- *Sequences of events* – interesting and complex interactions using basic actions.
- *Narrative split based on player actions* – the player can change the VE narrative.

We combined our benchmark interactions into three experimental tasks: two for the first session and one for the second. Each task built on the previous one, giving participants time to learn VRBridge and the domain. The tasks had to be specified at a level of detail that allowed comparison between participants on design activities, but also provided them with enough freedom to encourage exploration of narrative possibilities and creativity. Table 9.2 describes the tasks, showing the actions and interactions incorporated in each.

9.3.2. Task and VE descriptions

It was important that the look and feel of the VEs provided inspiring contexts for participants, in terms of the perceptual opportunities which a designer uses in creating meaningful interactions (see Section 4.3.3 for a description of perceptual opportunities). Participants were provided with a variety of sound files and objects for each task, gathered from free resources on the internet. We supplied content that would be recognisable

and interesting to a variety of participants. We provided small VEs, with visible limits, to reduce the complexity of the design task; participants could focus on interactions rather than exploring the environment.

Task	Focus of Task	Typical Actions	Benchmark Interactions
Task 1	Synchronised sound and actions	Object change of position and orientation; Object usage of sound and animation	Sequences of events
Task 2	Object movement and interaction	Object movement across a space avoiding obstacles; Object rotation on various axes; Object usage of sound and animation	Object reactions to player interaction; Sequences of events
Task 3	Player interaction, object movement and synchronised actions	Object movement across a space avoiding obstacles; Object rotation on various axes; Object usage of sound and animation	Object reaction to player interaction; Sequences of events; Narrative split based on player actions

Table 9.2: Abstract form of experimental tasks and the benchmark interactions incorporated in each.

Task 1

Task 1 focussed on creating a sequence of events to be synchronised with sound, incorporating animation and a condition to start the sequence. We created a garden space, populated with comic-style characters and objects. A screenshot of the VE is provided in Figure 9.1.

Figure 9.1: VE for Task 1, showing the garden scene and four characters.

We provided the following description to participants:

This VE is a garden with 4 characters: Hello Girl, Tigger, Chuck and Jack. Create a sequence of actions where Tigger starts his TigSong, and characters do actions during the song. Create at least 4 actions for Tigger, where starting his song is one, and at least 2 actions for 2 other characters. Remember that you are designing this sequence for a player to interact with. Try to make it fun to experience and natural looking.

Task 2

Task 2 focussed on object movement and rotation. We created an urban space, populated with action movie-style characters and objects. A screenshot of the VE is provided in Figure 9.2.

Figure 9.2: VE for Task 2, showing an industrial urban scene and three characters.

We provided the following description to participants:

This VE is a warehouse with 3 characters: John Maclane, Blade and an Assassin. Create actions where one character patrols the area, returning to where he started. Move him to at least one spot and one other character on his patrol, where he stops and does at least one action before moving on. The character that he stops at must also do at least one action when they meet. Then the patrol must repeat. Remember that you are designing actions for a player to interact with. Try to make them fun to experience and natural looking.

Task 3

Task 3 built on the first two, incorporating their interactions. It added the requirement of splitting the narrative based on player actions. We created a fantasy/science fiction space, populated with science fiction and fantasy characters and objects. A screenshot of the VE is provided in Figure 9.3.

Figure 9.3: VE for Task 3, showing a science fiction scene and six characters.

We provided the following description to participants:

This VE contains six characters. Create an interaction, where one character moves to the player, greets the player and then moves to the other characters. Now the action must split, depending on what the player does. If the player follows the first character, the others say nice things to the player. Then one of them must 'turn on the music' (hint: use sound system sounds for music and make the character do an action at the sound system). Then the characters must 'dance' (hint: piece together actions and movement). If the player does not follow the first character, the others insult the player. Remember that you are designing actions for a player to interact with. Try to make them fun to experience and natural looking.

Task 1 is the easiest and serves as a learning task. Task 2 requires understanding how to move objects in the space meaningfully. Task 3 is more free-form, as participants decide how extensive the dancing sequences should be. Only Task 3 explicitly requires that the player is involved in the narrative. However, participants are asked to provide player interaction in all tasks.

9.4. Measures

We have defined successful performance according to four criteria: accurate and skilful task completion; effective exploration; learning; and satisfaction. To evaluate these criteria, we developed three forms of data measurement: XML scores, computer logging and questionnaires. *XML scores* measure how well participants complete the tasks and explore VRBridge; *computer logging* shows the extent to which participants use VRBridge; and *questionnaires* measure participant satisfaction with VRBridge. How these results change over time provides us with information about learning. By triangulating the measures, we gain comprehensive evidence for our hypotheses.

9.4.1. XML scores

We saved participant output in separate XML files for each task, which we used to review the created VEs. We analysed the files according to a metric designed to elicit information about how well participants accomplished the design process. We analysed the following factors:

- Completion of basic task requirements (40 points);
- Exploration of VRBridge and the VR domain, including usage of Triggersets for specifying interactions (*variable sequencing* as they link in multiple ways); Timelines (*inevitable sequencing* as component actions always happen in the same order); VRBridge features; creativity in terms of narratively interesting design, effective usage of space and naturalness of interactions; and player freedom and involvement in interactions (60 points).

The metric measures three performance criteria: task completion; effective exploration; and learning. The items which evaluate task completion are described in Table 9.3 and those which evaluate exploration are described in Table 9.4. A more detailed description is provided in Appendix D. Learning is evaluated by comparing XML files. We designed the metric to measure the expertise with which participants created VR interactions and a rich experience for the player. Therefore, there is some overlap between the task-related items in Table 9.3 and the common items in Table 9.4, especially in the attention paid to narratively appropriate and creative interactions. However, Table 9.4 takes into account additional work and creativity which is not required by the tasks. In our results, we evaluate these separately as *Task Score* and *XML Explore Score* respectively.

Metric Item	**Description and Score**
Task 1 (40)	
Trigger Song	Start Tigger's song naturally and appropriately (4)
Tigger Actions	Provide 3 appropriate and synchronised actions for Tigger (16)
Other Actions	Provide 4 appropriate and synchronised actions for 2 other characters (20)

Metric Item	Description and Score
Task 2 (40)	
Patrol	Make a character patrol the area in a narratively interesting manner (8)
Return to Start	Make character return to start point (4)
Two Stops	Do 2 stops on patrol, one with another character (12)
Action at Stops	Do a narratively interesting and appropriate action at each stop (8)
Actor Response	Second character responds to patroller appropriately (4)
Repeat Patrol	Repeat patrol (4)
Task 3 (40)	
Greeting	Character greets player appropriately and returns to starting area (5)
Split Action	Action is split based on player actions in a narratively appropriate manner (8)
Nice Response	Multiple characters say synchronised and appropriate nice things to the player (4)
Nasty Response	Multiple characters say synchronised and appropriate nasty things to the player (4)
Turn on Music	Character does well timed, appropriate action near sound system and music plays (5)
Dance	Multiple characters do synchronised and varied actions when the music plays, which simulate dancing well (14)

Table 9.3: Metric used to evaluate completion of task requirements for all three experimental tasks.

Table 9.3 shows how we define skilful and accurate *task completion* for each task. A participant who scores higher on this metric has completed the task in a more expert manner, by accomplishing the technical requirements for effective interactions and creating interesting narratives.

Metric Item	Description and Score (60 Overall)
Variable Sequencing	Effective usage and exploration of Triggersets, including individual triggers, actions and conditions. Multiple, linked Triggersets with varied components that are appropriate (20).
Inevitable Sequencing	Effective usage and exploration of Timelines, using multiple objects and synchronised actions, which are added meaningfully to Triggersets (10).
Features	Effective usage of multiple features: waypoints, locations, and the Object Panel for changing details. Used meaningfully and interestingly, e.g., waypoints mark a route (5).
Creativity	Interesting design, appropriate usage of VE space (e.g., trigger distances) and natural interactions (e.g., appropriately synchronised animations and sounds, correct facing) (15).
Player Freedom	Freedom of movement provided to the player, and appropriate inclusion in the VE in multiple ways (e.g., player input, player proximity, objects turn to face player) (10)

Table 9.4: Metric used to evaluate exploration of VRBridge and the VR domain.

Table 9.4 shows how we define *effective exploration* of both VRBridge and the domain of VR. The total *XML Explore* score measures *general expertise* in customising and using VRBridge and working in VR. Participants explored VRBridge more successfully if they achieved the following:

- *Customised Triggersets*: Triggersets linked non-linearly, comprising a variety of triggers, conditions and actions customised for more accurate and interesting interactions.
- *Synchronised Timelines*: Timelines with multiple objects and synchronised actions, linked to the Triggersets.
- *Customised space*: Locations and waypoints placed to create accurate interactions and exploit the space.
- *Customised Objects*: Using the Object Panel to customise object details for narrative effect.

Participants explored the domain of VR more successfully if they demonstrated expertise in authoring, according to the following criteria:

- *Player freedom*: Interactions include the player and provide freedom of movement and the ability to make choices with consequences.
- *Exploited space*: Using the space of the VE well, so it can be explored by the player; appropriate distances used in interactions.
- *Interesting narratives*: Narratives which show awareness of player experience.
- *Natural interactions*: Natural-looking interactions, e.g., objects performing animations while moving and turning to face the direction of movement.

The XML files provide three consecutive sets of scores for each participant, which evaluate *learning*. The most useful items are *XML Explore* scores, as they are less affected by the nature of the different tasks and focus on general domain-relevant skill and usage of VRBridge. The *Task Scores* are also useful to evaluate learning, but are affected by the nature of the tasks: Task 1 was easier than the others and had external support from the experimenter as it was a learning task; and more time was available for Task 3.

For group comparison, the *Task Scores* are important as they measure how well participants performed required tasks. Differences in creativity and interest will impact more on the *XML Explore* scores. Sections of the *XML Explore* score provide data on differences in exploratory activities fostered by the design aids: player freedom provided; exploration of Triggersets versus Timelines; usage of features; and creativity.

9.4.2. Computer logging

We recorded detailed computer logs of VRBridge usage in both sessions. We calculated the overall amount of time each representation was open, for an indication of how much they were referenced (combined with the number of manipulations and number of times opened). For scaffolding, we were particularly interested in how it was faded. We also recorded system crashes, for a measure of data reliability. Table 9.5 presents

the logged items, organised according to work that produced output or gathered information, and time-based usage. The table links the logged items to exploratory activities, using the terms defined for *effective exploration* above. By combining the logging results, which measure amount of usage, with the *XML Explore* results, which measure successful usage, we gain a more detailed understanding of exploration. By analysing our logs, we can explore group differences in usage initially, and differences in *learning* by comparing changes in usage across sessions.

Logging Items	**Links to Effective Exploration**
Producing output	
Triggerset authoring: creating and editing components	*Customised Triggersets*
Object Panel manipulation: changing any details	*Customised Objects*
Timeline creation: creating and opening Timelines	*Synchronised Timelines*
Timeline authoring: adding, editing and moving actions	*Synchronised Timelines*
Waypoint creation	*Customised space*; *Exploited space*
Location creation	*Customised space*; *Exploited space*
Floorplan opened	*Exploited space; Customised Objects*
Floorplan manipulations	*Exploited space; Customised Objects*
Rotation wizard used	*Natural interactions*
Movement wizard used	*Natural interactions*
Gathering information	
Triggerset manipulations using context menus	*Customised Triggersets*
Locations and waypoints viewed in 3D Window	*Customised space*; *Exploited space*
Pausing the 3D World	*General expertise*
Sound previews viewed	*Interesting narratives*
Animation previews viewed	*Interesting narratives*
Sequence Diagram opened	*Customised Triggersets*; *Player Freedom*
Sequence Diagram manipulations	*Customised Triggersets; Player Freedom*
Run Mode started	*General expertise*
Context Sensitive Hints used in total	*General expertise*
Context Sensitive Hints used per screen (InterPlay, Main Menu, 3D Window, Sequence, Floorplan)	*General expertise*
Time-based usage	
Sequence Diagram viewing time in minutes	*Customised Triggersets; Player Freedom*
Floorplan viewing time in minutes	*Exploited space; Customised Objects*

Logging Items	Links to Effective Exploration
Run Mode viewing time in minutes	*General expertise*
Tooltips turned off or on from Main menu	*General expertise*

Table 9.5: Features that were logged during the evaluation, showing how they relate to exploration.

9.4.3. Questionnaires

We used five questionnaires in this study, four of which we created (these are reproduced in Appendix E). A *demographic questionnaire* elicited information about gender, age, study level, university department, and computer and programming experience (Appendix E1). For all of the others, we followed questionnaire design guidelines (Anastasi 1996) to maximise construct validity (the extent to which the questionnaire measures what it claims to measure). We also conducted reliability and validity tests: Cronbach's alpha to measure reliability and average inter-item correlation to measure validity. The results of our validation showed that all of the questionnaires, and their sub-sections, were reliable and valid. These are reported below and described in more detail in Appendix E5.

The *QUIS 7 questionnaire*[5] was developed and validated by the University of Maryland. It comprises twelve sections with questions on a 9-point Likert scale, each of which measures an aspect of system usability. Inappropriate sections can be removed without lessening its validity and reliability. Our QUIS 7 questionnaire had 59 items and retained sections on overall reactions, screens, terminology and learning. We removed sections on length of system experience, past experience, technical manuals, on-line tutorials, multimedia, teleconferencing, system capabilities and software installation. Its Cronbach's alpha was 0.95 and its average inter-item correlation was 0.26, both of which fall into acceptable ranges (Cronbach 1990, Anastasi 1996). The correlation value is marginal but this is expected for questionnaires which measure multi-dimensional concepts.

We devised three additional questionnaires specifically about VRBridge, following the QUIS 7 format for consistency.

- The *System questionnaire* (Appendix E2) concerns general VRBridge features and comprises 18 items. It has two main sections focussing on Triggersets and Timelines. We validated the Triggerset and Timeline sections as well as the full questionnaire, so that we could analyse satisfaction with these features, as well as overall satisfaction. We obtained a Cronbach's alpha of 0.87, 0.9 and 0.94 respectively; and an average inter-item correlation of 0.3, 0.64 and 0.79 respectively. Since QUIS is validated, we tested the correlation between it and the System Questionnaire to measure concurrent validity of the System

[5] *http://lap.umd.edu/quis/*

Questionnaire, gaining a significant correlation coefficient of 0.84. The *System questionnaire* was the most problematic questionnaire: even though we reduced its size from 21 to increase its internal validity, both its reliability and validity were still marginal. However, they fell within acceptable values, and increased for the sub-sections, and so we kept the questionnaire.

- The *Representations questionnaire* (Appendix E3) was given to those groups which received representations, and comprises 16 items. It has sections on the Floorplan, Sequence Diagram and Run Mode. We obtained a Cronbach's alpha of 0.92, 0.93, 0.94 and 0.9 respectively; and an average inter-item correlation of 0.45, 0.68, 0.7 and 0.75 respectively.
- The *Scaffolding questionnaire* (Appendix E4) was given to those groups which received scaffolding, and comprises 22 items. It has sections on Context Sensitive Hints, Tooltips and Wizards. We obtained a Cronbach's alpha of 0.97, 0.96, 0.93 and 0.98 respectively; and an average inter-item correlation of 0.65, 0.93, 0.75 and 0.86 respectively.

9.5. Participants

The target group for this research is broad. The system should be useful to people who work with computers but are not necessarily programmers. Therefore, a wide cross-section of university students, from different departments and years of study, is an appropriate population from which to draw a sample. We advertised using online course management systems of different departments, including English, Architecture, Mathematics, Engineering, Commerce and Psychology. The advertisement emphasised that the experiment involved creating dynamic 3D worlds, and that we preferred no programming experience. Participants received a small financial incentive for completing the experiment. Ultimately, 149 participants signed up, of whom only three did not return for the second session.

All participants were between the ages of 18 and 23, 56% of the sample was female, and 5% of the participants were postgraduates, with the rest spread fairly evenly through undergraduate years. 52% were Humanities students, with 38% of those being Psychology majors; 31% were Science students, with 8% of those being Computer Science majors.

We removed the results of those who indicated they could program. Although our target group does not exclude programmers, the primary focus of VRBridge is supporting novices. Our analysis would yield less ambiguous results if we controlled for programming experience. Eighteen participants or 12% of the original sample were removed in this way. We also removed the results of four participants who had experienced severe system crashes. Overall, we removed 22 observations, or about 15% of the original sample. We

therefore conducted our analyses with a sample of 124 participants, comprising 31 data points in each group. Table 9.6 presents the final distribution of participants by field of study.

Faculty/Department	Number of Participants	Percentage of Total
Humanities excluding Psychology	49	39.5
Psychology	20	16.1
Science	29	23.4
Commerce	12	9.7
Engineering	5	4
Architecture	9	7.3
Total	**124**	**100**

Table 9.6: Breakdown of participants by field of study, showing actual numbers and percentages of the total.

9.6. Procedure

This evaluation was a longitudinal experiment: participants signed up for two sessions, one week apart. The first session took one-and-a-half hours and the second took one hour. Each session slot comprised four participants. They were not aware of different groups, having been told that we were evaluating a system for creating VE actions without programming. To control awareness of groups, all participants in a single slot were given the same version of VRBridge. Therefore, group assignment was based on the sessions for which participants signed up, which served to randomise it. We cycled through groups, ensuring that the total number of participants in each group remained equal. Four desktop computers with similar specifications were set up in a dedicated experiment room with 17" monitors and headphones. They were arranged so that participants could not see the screens of other computers. An experimenter was present at all times.

9.6.1. Pilot study

We conducted a small pilot study with eight participants to practice the experimental procedure, test the task suitability, and validate our measures. We found that task complexity was suitable; all participants achieved results in the allotted time. We refined our experimental procedure, introduction and tutorial. Participants frequently asked for reminders while working. Therefore, we created a simple booklet with a description of VR and how to use VRBridge (see Appendix F). We also decided that the experimenter would answer technical questions during Task 1. This created a smoother learning curve, as participants had a tutorial, an individual task with technical assistance, and then two tasks to accomplish alone.

We also evaluated our XML analysis metric, and found that it was sufficiently detailed to capture feature usage and exploratory actions. It was tested for reliability by two independent raters who were unaware of

participant groups. An inter-rater agreement of 98% was reached, which we found satisfactory. Our analysis for the full experiment used double-blinding (i.e., both participants and markers were unaware of group).

9.6.2. Session 1

Participants were provided with the VRBridge booklet for reference during the tasks. They received a 15 minute VRBridge introduction and tutorial on the system, during which we described VR and emphasised the role of the player. Thereafter, we distributed the Task 1 description and loaded its XML file. This contained the world geometry and all objects, animations and sounds. Participants had half-an-hour to work on each task. They only received technical assistance during Task 1, so that the scaffolding of the system itself could be evaluated. After half-an-hour, participants saved their work, received the Task 2 description and loaded its file. At the end of the session, they received the Demographic, QUIS 7 and System questionnaires. The Representation and Combination groups received the Representations questionnaire and the Scaffolding and Combination groups received the Scaffolding questionnaire.

9.6.3. Session 2

When participants returned for the second session, they again received the VRBridge booklet, and we reiterated the importance of designing for the player. Then we gave participants the Task 3 description and loaded its XML file. They had forty-five minutes for this task, since it was more complicated and longer than the first two. Thereafter, participants were given the same questionnaires as before, except for the demographic questionnaire. Then they were paid and thanked.

9.7. Results

All of our analyses were conducted using Microsoft Excel and StatSoft Statistica. We evaluated the data for differences in effects correlating with sample demographics (such as study level, department or gender), and found none. Below, we present significant results organised by the data measures. Tables of descriptive statistics for all measures are presented in Appendix G. The standard deviations of all dependent variables are quite high, which was expected as the experiment required creative work by participants, which is subject to large individual differences.

9.7.1. Task score analysis

We conducted two analyses on the XML files. The metric scored participants on completion of task requirements and exploration of VRBridge and the domain (see Section 9.4.1). Here, we describe the task score analysis. We conducted a one-way multivariate analysis of variance (ANOVA, with multiple dependent variables), with Group as the independent variable (IV), and the three task scores as dependent

variables (DVs). See Table G1 in Appendix G for descriptive statistics. Our investigation revealed a significant effect: $F(9, 287.33) = 1.9617$, $p < 0.05$. A means plot of this is presented in Figure 9.4.

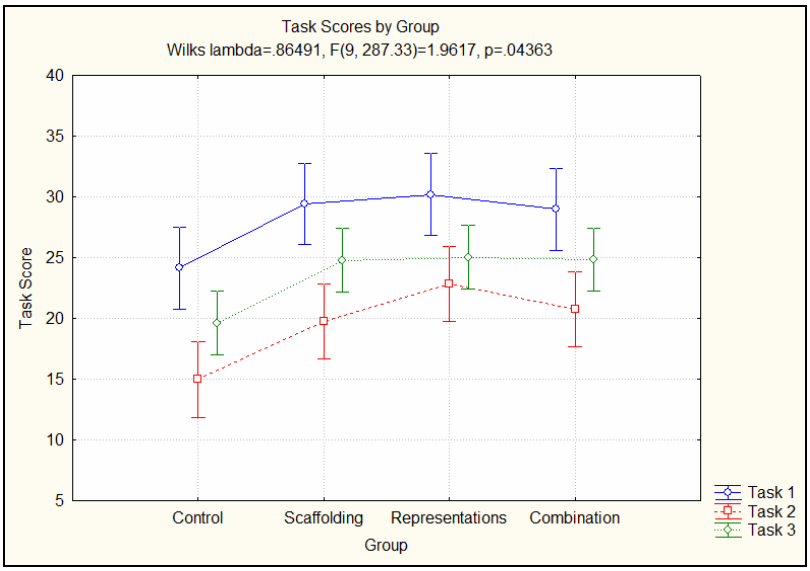


Figure 9.4: Means plot showing the effect of scaffolding and representations (i.e., Group) on task scores.

Scores cannot be compared across tasks, as they provide different levels of difficulty. The pattern of scores across Group is similar for each task: all groups scored best in the Task 1 and worst in Task 2. We conducted post-hoc analyses using Fisher's LSD test and determined that the Control group scored significantly less than the other groups in all three tasks: $p < 0.05$, $p < 0.05$ and $p < 0.01$ for Scaffolding; $p < 0.05$, $p < 0.001$ and $p < 0.01$ for Representation; and $p < 0.05$, $p < 0.01$ and $p < 0.01$ for Combination. There were no other significant differences.

9.7.2. XML Explore analysis

We conducted two analyses to investigate VRBridge and domain exploration. The first investigated the *XML Explore Total*; the second investigated components of *XML Explore* described in Table 9.4. These results can be compared across task, as they are designed to be task-independent, focussing on the expertise with which participants explored. See Table G2 in Appendix G for descriptive statistics.

XML Explore Total

We conducted a factorial mixed-design ANOVA, with Group as the between-subjects IV, Task as the repeated measures IV, and *XML Explore Total* as the DV. Our analysis revealed no significant main effect for Group ($p = 0.096$) and no interaction effect ($p = 0.11$), but a highly significant Task effect: $F(2,240) = 44.472$, $p < 0.00001$. The plot for *XML Explore Total* across Group and Task is presented in Figure 9.5.

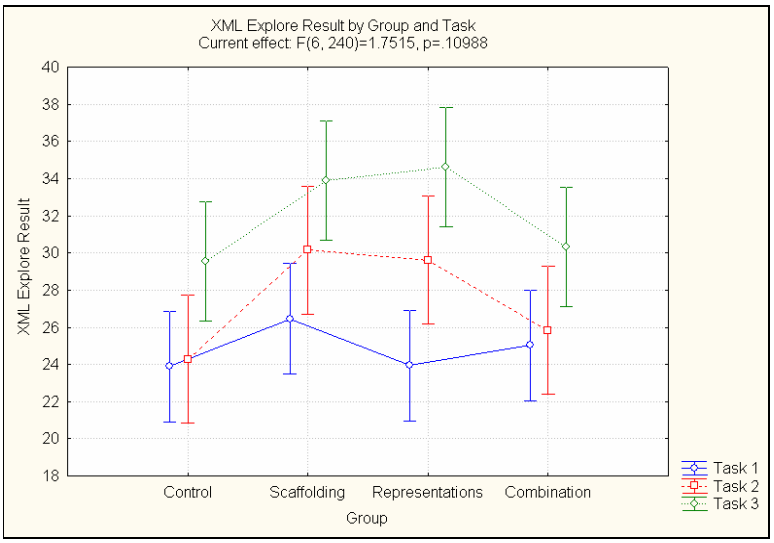


Figure 9.5: Means plot showing the effects of Group and Task on XML Explore scores.

The pattern of scores for Task 1 is different to Tasks 2 and 3. This would hide group differences, as the tasks are combined for the analysis. Because the overall results were not significant, we used the Scheffe post-hoc test to analyse the effect of Group on *XML Explore Total* separately for each task. This is a very conservative post-hoc test (meaning that it is more likely to incorrectly reject a significant result than incorrectly accept a non-significant result), which is why it can be used when there is no overall significant result.

In Task 1, there were no significant differences. In Task 2, the score for the Control group was significantly lower than those for the Scaffolding and Representation groups ($p < 0.05$ for both). In Task 3, the score for the Control group was significantly lower than that for the Representation group and the difference between the Control and Scaffolding groups approached significance ($p < 0.05$ and $p = 0.059$ respectively). The Combination group scores were much lower than those for the Representation and Scaffolding groups in Tasks 2 and 3, but the differences were not significant.

We also conducted post-hoc tests to determine the nature of the significance for the Task effect. We found that the *XML Explore* scores were highly significantly different to each other. The score for Task 2 was significantly higher than that for Task 1 ($p < 0.001$), and the score for Task 3 was significantly higher than those for both other tasks ($p < 0.0000001$ for both).

XML Explore Components

We conducted a factorial mixed-design *multivariate* ANOVA, with Group as the between-subjects IV, Task as the repeated measures IV, and *Variable Sequencing*, *Inevitable Sequencing*, *Creativity*, *Features* and *User (Player) Freedom* as DVs. Our analysis revealed a highly significant main effect for Group ($F(15,320.63) =$

3.958, $p < 0.000001$) and Task ($F(10,111) = 23.229$, $p < 0.00001$), but no significant interaction effect ($p = 0.096$). The means plot for the effect of Group and Task on *XML Explore* components is presented in Figure 9.6. We conducted post-hoc LSD tests to determine the nature of the significant Group effect:

- *Variable Sequencing*: the Control group scored significantly lower than all other groups: $p < 0.05$ for Scaffolding; and $p < 0.01$ for Representation and Combination.
- *Inevitable Sequencing*: the Scaffolding group scored significantly higher than the Combination group ($p < 0.05$), which scored lowest in all three tasks.
- *Creativity*: the Scaffolding and Representation groups scored significantly higher than the Control group: $p < 0.01$ and $p < 0.05$ respectively. In Tasks 2 and 3, the Combination group scored much less than the Scaffolding and Representation groups, but this result is not significant.
- *Features*: the Representation and Combination groups scored highly significantly more than both the Control and Scaffolding groups: $p < 0.00001$ for the Control group; and $p < 0.01$ for the Scaffolding group. The standard deviations for this component are very large, even allowing for creative differences. This is due to the fact that many participants did not use these features at all and therefore scored zero, skewing the variable's distribution. However, as it yields highly significant results, we retain it.
- *User (Player) Freedom*: the Control group scored significantly less than all three of the other groups: $p < 0.05$ for Scaffolding; $p < 0.01$ for Representation; and $p < 0.01$ for Combination.

We also conducted post-hoc tests to determine the nature of the significant Task effect.

- *Variable Sequencing*: the score for Task 3 was highly significantly increased from those for Tasks 1 and 2 ($p < 0.0000001$ for both), while the difference in score between Task 1 and Task 2 approached significance ($p = 0.0581$), with Task 2 being higher.
- *Inevitable Sequencing*: the score for Task 3 was highly significantly increased from those for Tasks 1 and 2 ($p < 0.0000001$ for both).
- *Creativity*: the score for Task 2 was significantly higher than that for Task 1 ($p < 0.001$); and the score for Task 3 was significantly higher than those for both previous tasks ($p < 0.0000001$ for both).
- *Features*: the score for Task 2 was highly significantly increased from those for Tasks 1 and 3 ($p < 0.0000001$ and $p < 0.0001$ respectively); and the score for Task 3 was significantly higher than that for Task 1 ($p < 0.05$).
- *User (Player) Freedom*: there were no significant differences between tasks.

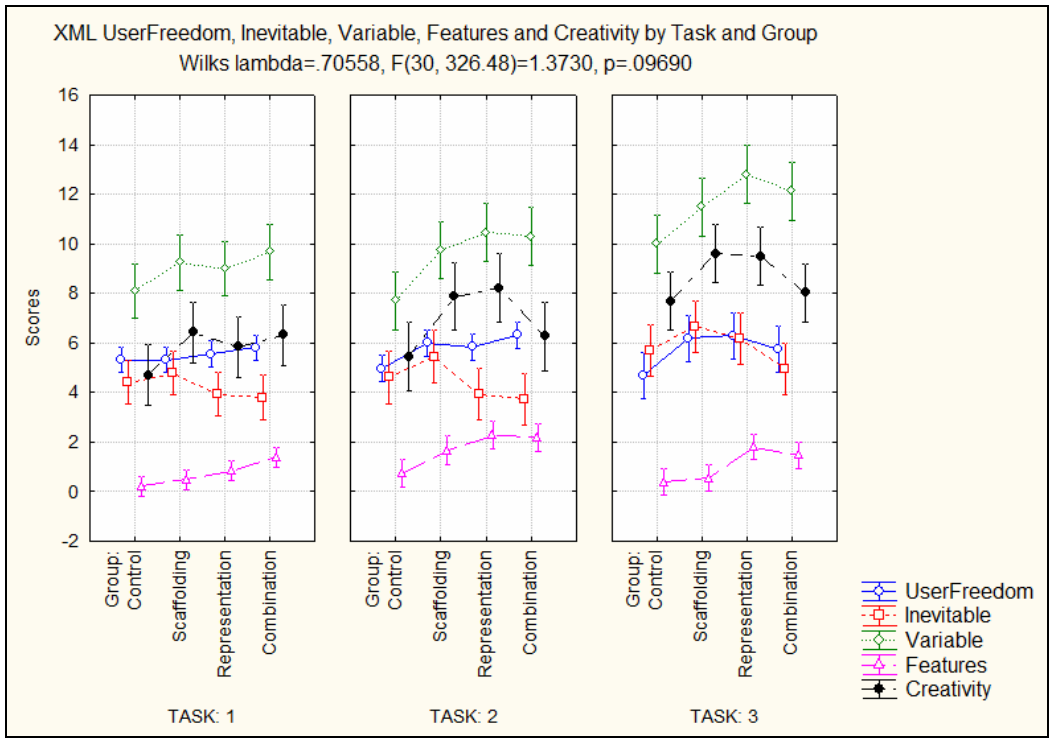


Figure 9.6: Means plot showing the effect of Experimental Group and Task on XML Explore component scores.

9.7.3. Logging results

Our logging results did not yield any significant results because of the nature of the data. For these variables, the difference in means is small relative to the variance. Most of these variables have a large dispersion, with large standard deviations, high maximums and minimums of zero, showing that participants worked in very different ways with VRBridge. However, they can still yield interesting descriptive statistics which provide insight into how participants in different groups worked with VRBridge and the design aids. We report all mean comparisons with confidence intervals to provide an indication of reliability. We discuss the statistics separately for variables that apply to all groups and those that only apply to groups with representations and/or scaffolding. Below, we present and discuss column charts of these variables (see Table 9.5 for a complete list).

General VRBridge Functions

See Table G3 in Appendix G for descriptive statistics on logging variables for general VRBridge functions, divided into those used for authoring and those used for gathering information. Figure 9.7 displays the average number of *Triggerset and Timeline authoring actions*, and *Timelines created or opened* per Group and Visit. These variables cover the primary VRBridge authoring mechanisms. Triggersets and Timelines have opposite usage patterns. The Representation and Combination groups authored the most Triggersets in Visit 1 (Mean 32.74, Confidence Interval (CI) 6.73 and Mean 30.55, CI 5.97 respectively); in Visit 2 the

Scaffolding group authored slightly more than the Combination group (Mean 23.68, CI 4.47 and Mean 23.35, CI 5.46 respectively). The Control group authored the fewest Triggersets in both visits and its maximum values were much lower (Mean 22.42, CI 4.22 and Mean 17.1, CI 4.01 respectively). The Scaffolding and Control groups used the most Timelines in both visits (For TimeCreate: Mean 7.58, CI 2.9 and Mean 6.74, CI 2.62 respectively in Visit 1 and Mean 7.58, CI 2.68 and Mean 6.58, CI 3.02 respectively in Visit 2; For TimeAuth: Mean 16.81, CI 8.6 and Mean 15.9, CI 8.54 respectively in Visit 1 and Mean 16.68, CI 6.79 and Mean 14.68, CI 6.66 respectively in Visit 2) and the Representation group used the least (For TimeCreate: Mean 4.13, CI 1.78 in Visit 1 and Mean 5.23, CI 1.95 in Visit 2; For TimeAuth: Mean 8.52, CI 4.51 in Visit 1 and Mean 12.58, CI 7.59 in Visit 2). These differences decreased in Visit 2.

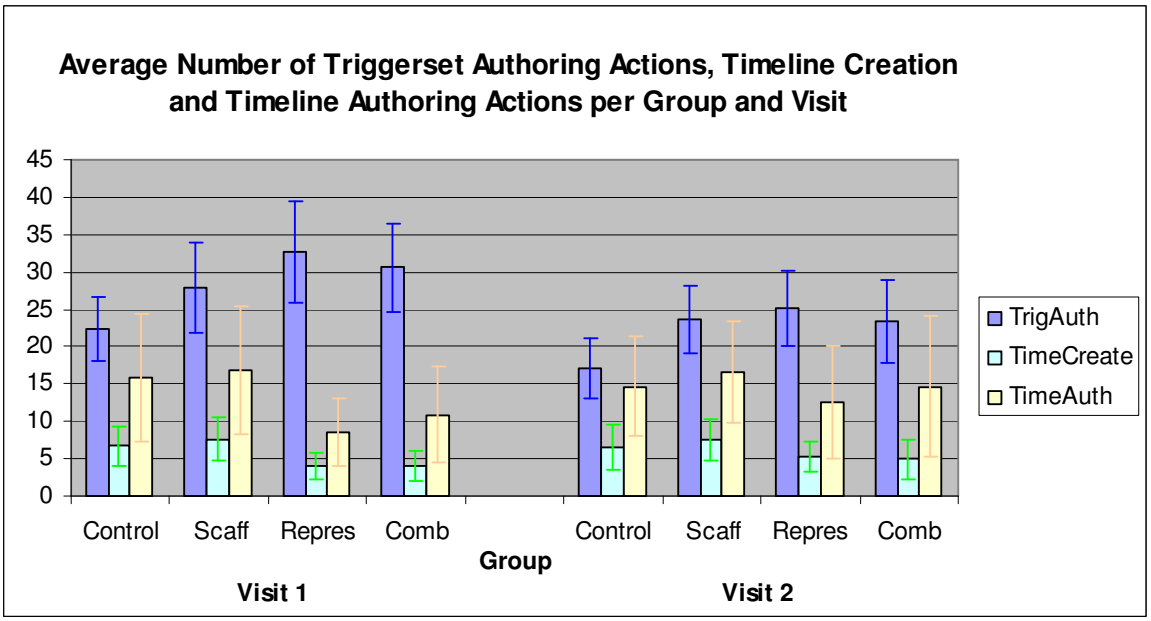


Figure 9.7: Column chart of the average number of Triggerset authoring actions (TrigAuth), Timelines created or opened (TimeCreate) and Timeline authoring actions (TimeAuth) for each Group and Visit. Vertical bars show confidence intervals.

Figure 9.8 displays the average number of *Object Panel manipulations*, and *waypoints and locations created* per Group and Visit. These variables cover the primary mechanisms for working with the space and objects of the VE. In Visit 1, the pattern of usage is the same as that for Triggersets: the Representation and Combination groups performed many more actions than the other groups (For ObjPnlMan: Mean 9.03, CI 3.23 and Mean 7.61, CI 3.51 respectively; for WayCreate: Mean 1.87, CI 1.01 and Mean 2.48, CI 1.07 respectively; for LocCreate: Mean 0.94, CI 0.89 and Mean 0.81, CI 0.91 respectively). In Visit 2, this pattern continues for waypoints and locations, but the Scaffolding group performed more Object Panel manipulations than the Representation group (Mean 5.16, CI 6.95 and Mean 3.81 CI 1.99 respectively).

While all other groups decreased their manipulations in Visit 2, those of the Scaffolding group remained the same (although its confidence interval increased dramatically from 3.69 to 6.95). The Control group performed by far the fewest manipulations in both visits. The Scaffolding and Control groups created few waypoints and locations in both visits.

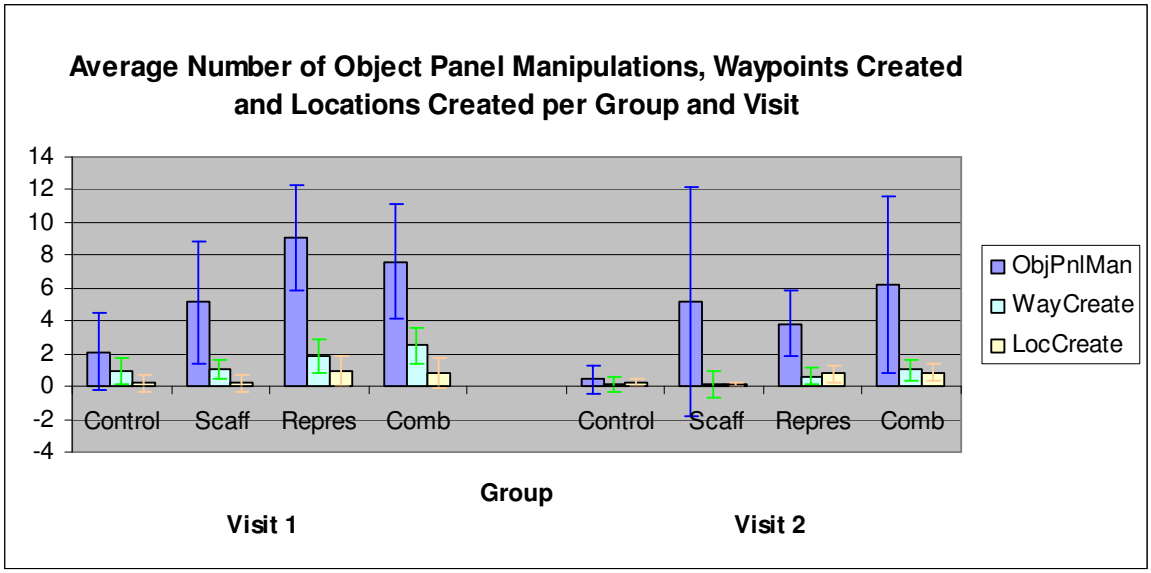


Figure 9.8: Column chart of the average number of Object Panel manipulations (ObjPnlMan), waypoints created (WayCreate) and locations (LocCreate) created, for each Group and Visit. Vertical bars show confidence intervals.

Figure 9.9 displays the average usage of *sound and animation previews*, *Triggerset manipulations*, *3D pauses* and number of *locations and waypoints viewed in 3D* per Group and Visit. These variables cover the mechanisms available to all participants for gathering information about their interactions in 3D space and the narrative possibilities of the content. The Scaffolding and Control groups previewed the most animations in both visits (the Scaffolding group by a considerable margin in Visit 1) (Mean 8.1, CI 6.01 and Mean 5.16 CI 3.08 respectively in Visit 1; and Mean 6.68, CI 6.35 and Mean 7.77 CI 7.04 respectively in Visit 2), while the Scaffolding and Combination groups previewed the most sounds (Mean 6.45, CI 3.54 and Mean 4.84 CI 2.98 respectively in Visit 1; and Mean 8.45, CI 5.15 and Mean 8 CI 3.95 respectively in Visit 2).

The average Triggerset manipulations were similar for all groups. The Representation group was highest and the Scaffolding group lowest in both visits (Mean 4.1, CI 1.64 and Mean 2.94 CI 1.31 respectively in Visit 1; and Mean 4.32, CI 1.37 and Mean 3.39 CI 1.29 respectively in Visit 2). In Visit 2, the Combination group's manipulations increased the most. Participants did not pause the 3D world often and barely viewed locations and waypoints. In both visits the Representation group paused most, but in Visit 2, the Combination group

paused a similar amount (Mean 2.74, CI 1.31 and Mean 2.03 CI 0.93 respectively in Visit 1; and Mean 1.32, CI 0.96 and Mean 1.29 CI 1.22 respectively in Visit 2). In both visits, the Control group paused the least; its maximum values were less than half those of the Scaffolding group, the next lowest (Mean 1.1, CI 0.37 in Visit 1; and Mean 0.39, CI 0.2 in Visit 2). The Scaffolding and Control groups viewed the most locations and waypoints in both visits (Mean 1.19, CI 0.87 and Mean 0.58 CI 0.47 respectively in Visit 1; and Mean 0.35, CI 0.63 and Mean 0.26 CI 0.44 respectively in Visit 2).

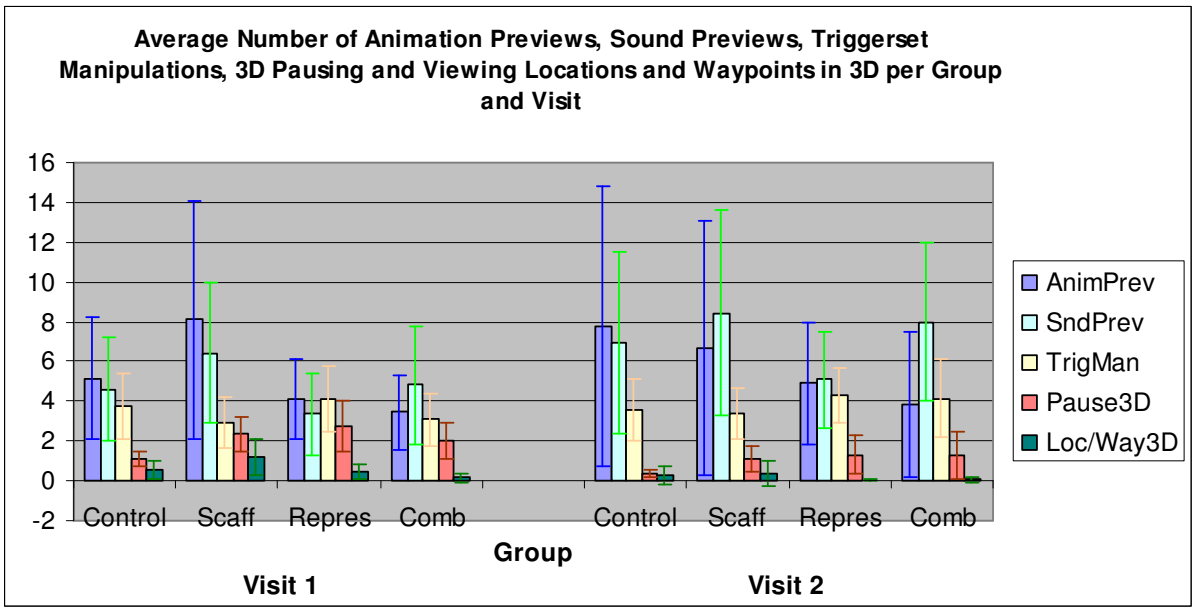


Figure 9.9: Column chart of the average number of sound (SndPrev) and animation previews (AnimPrev) used, Triggerset manipulations (TrigMan), 3D pausing (Pause3D) and viewing locations and waypoints in 3D ()LocWay3D), for each Group and Visit. Vertical bars show confidence intervals.

Floorplan, Sequence Diagram and Run Mode

Table G4 in Appendix G presents descriptive statistics for the representation logging variables. In general, the dispersion of these variables is not large, although differences in viewing times and their standard deviations are very large. All except *Floorplan Open* have minimums of zero.

Figure 9.10 displays the average number of times the *Floorplan was opened and manipulated*, and its *average viewing time* per Group and Visit. The Floorplan helps the designer to customise and gather information about object spatial configurations and the VE space.

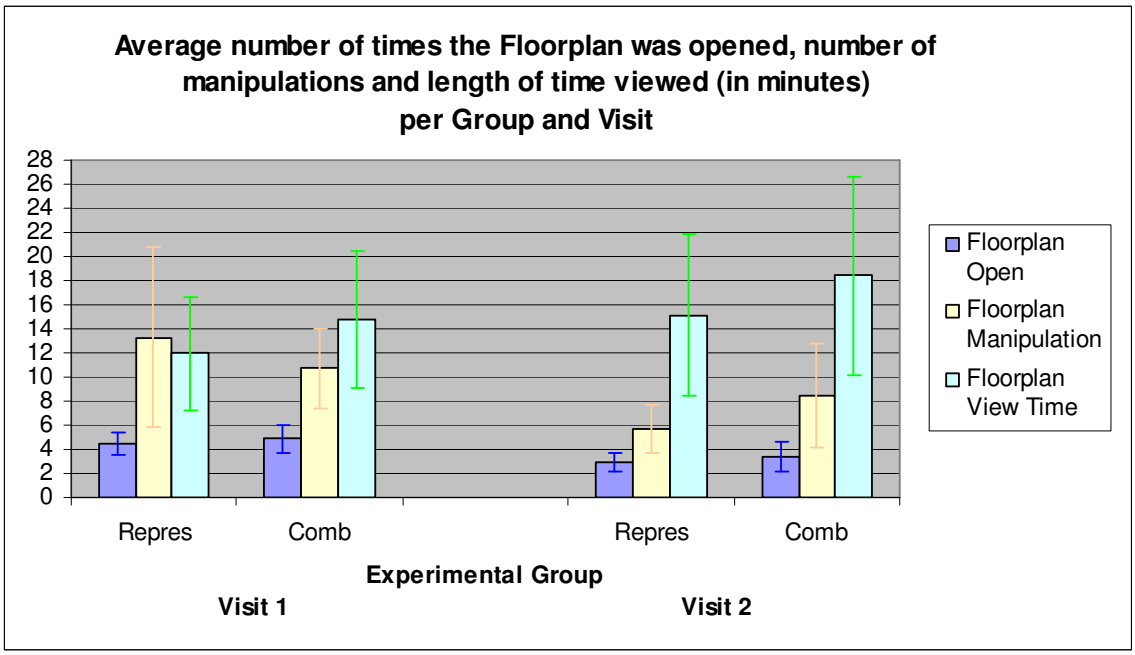


Figure 9.10: Column chart of the average number of times the Floorplan was opened (Floorplan Open) and manipulated (Floorplan Manipulation), and the average viewing time (Floorplan View Time), for each Group and Visit. Vertical bars show confidence intervals.

The Combination group opened the Floorplan slightly more and viewed it longer than the Representation group in both visits (For Floorplan Open: Mean 4.87, CI 1.2 and Mean 4.45 CI 0.95 respectively in Visit 1; Mean 3.35, CI 1.28 and Mean 2.94 CI 0.77 respectively in Visit 2. For Floorplan View Time: Mean 14.46, CI 5.69 and Mean 11.56 CI 4.65 respectively in Visit 1; and Mean 18.24, CI 8.21and Mean 15.07 CI 6.7 respectively in Visit 2). In Visit 1, the Representation group manipulated more than the Combination group, but this reversed in Visit 2 (Mean 13.29, CI 7.46 and Mean 10.71 CI 3.34 respectively in Visit 1; Mean 5.68, CI 1.96 and Mean 8.48 CI 4.27 respectively in Visit 2). The viewing time for both groups increased in Visit 2 (although the other variables decreased).

Figure 9.11 displays the average number of times the *Sequence Diagram was opened and manipulated*, and its *average viewing time* per Group and Visit. The Sequence Diagram helps the designer to gather information for authoring and linking Triggersets more expertly; and understand the role of the player in the interactions. The Combination and Representation groups opened the Sequence Diagram a similar number of times in both visits (Mean 3.39, CI 0.94 and Mean 3.13 CI 0.67 respectively in Visit 1; Mean 1.16, CI 0.4 and Mean 1.71 CI 0.7 respectively in Visit 2). Their manipulations were also close in Visit 1, but the Representation group did over twice as many manipulations as the Combination group in Visit 2 (Mean 3.74, CI 0.59 and Mean 3.23 CI 0.58 respectively in Visit 1; Mean 2.58, CI 0.55 and Mean 0.9 CI 0.27

respectively in Visit 2). The Combination group viewed the Sequence Diagram for much longer than the Representation group in both visits (Mean 10.13, CI 3.67 and Mean 5.33 CI 1.94 respectively in Visit 1; Mean 5.31, CI 4.03 and Mean 2.36 CI 1.07 respectively in Visit 2). All variables decreased in Visit 2. If we compare Floorplan and Sequence Diagram usage, we see that, although they were opened an equivalent number of times, the Floorplan was manipulated more and viewed for longer in both visits.

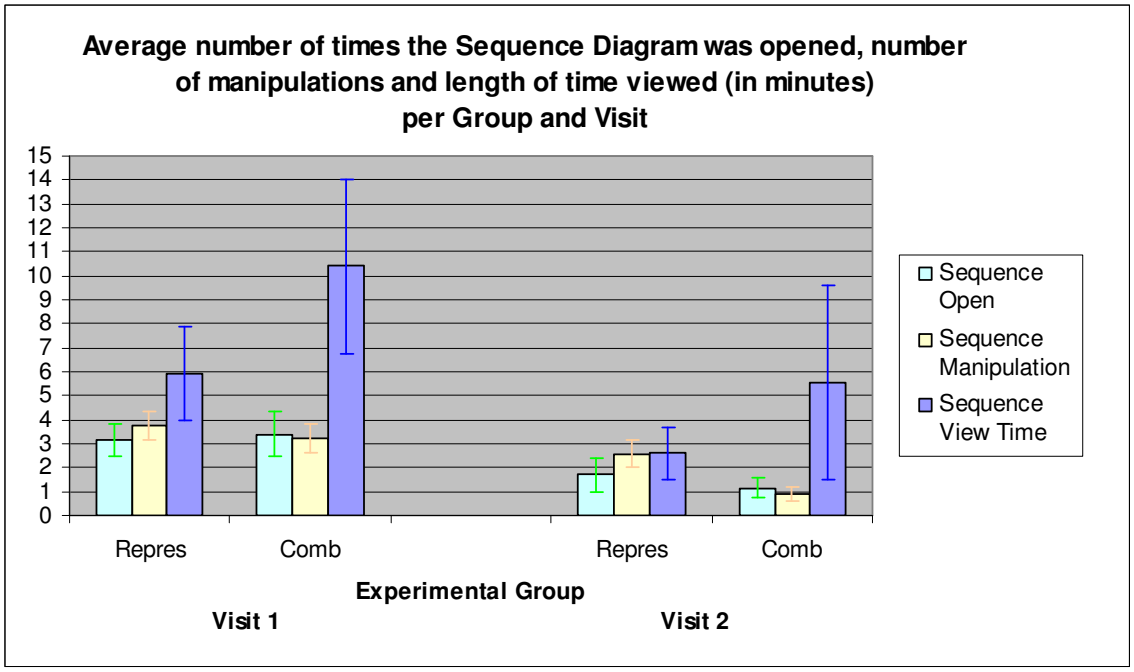


Figure 9.11: Column chart of the average number of times the Sequence Diagram was opened (Sequence Open) and manipulated (Sequence Manipulation), and the average viewing time (Sequence View Time) for each Group and Visit. Vertical bars show confidence intervals.

Figure 9.12 displays the average number of times the *Run Mode was started* and its *average viewing time* per Group and Visit. The groups started the Run Mode a similar number of times in Visit 1, but the Representation group started it three times more often than the Combination group in Visit 2 (Mean 3.1, CI 1.39 and Mean 2.97 CI 1.36 respectively in Visit 1; Mean 1.68, CI 1.07 and Mean 0.56 CI 0.79 respectively in Visit 2). The Representation group viewed the Run Mode longer than the Combination group in Visit 1, but this reversed in Visit 2 (Mean 5, CI 1.48 and Mean 2.27 CI 0.82 respectively in Visit 1; Mean 2.38, CI 1.2 and Mean 4.25 CI 1.13 respectively in Visit 2). The Combination group increased its viewing time in Visit 2, while decreasing the number of times started.

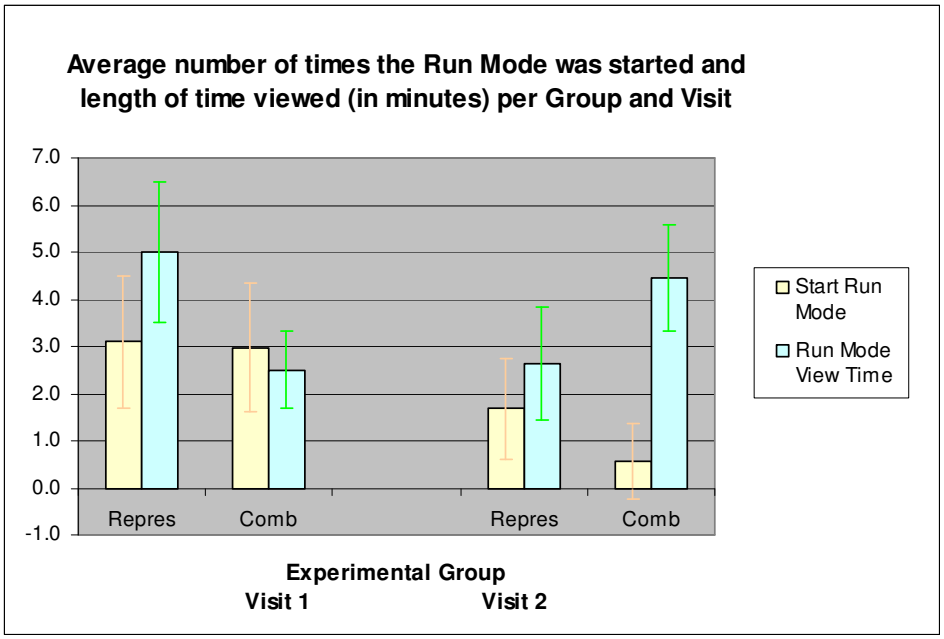


Figure 9.12: Column chart of the average number of times the Run Mode was started (Start Run Mode) and the average viewing time (Run Mode View Time) for each Group and Visit. Vertical bars show confidence intervals.

Scaffolding: Tooltips, Context Sensitive Hints and Wizards

Table G5 in Appendix G presents descriptive statistics of the scaffolding logging variables. All of the variables have a large dispersion, with large standard deviations compared to their means, relatively high maximums and minimums of zero (except *3D Window Context Hints* and therefore *Total Hints* for the Scaffolding group). For *Tooltip* usage, we logged whether participants turned them off. Only two participants did this, both from the Combination group in Visit 2. The graphs show that participants did not use the scaffolding very often. However, the Tooltips were always present on the interface. Therefore, all participants who received scaffolding were exposed to the Tooltips constantly.

Figure 9.13 displays the average usage of the *Rotation and Movement Wizards*, and *Total Context Sensitive Hints* per Group and Visit, and Figure 9.14 displays the average usage of *Context Sensitive Hints* from each screen. The Wizards are useful for increasing expertise in 3D interaction authoring, especially creating more natural interactions. The Context Sensitive Hints and Tooltips are useful for increasing domain and tool knowledge and expertise. The Scaffolding group used the Wizards much more than the Combination group (Rotation Wizard: Mean 1.03, CI 0.91 and Mean 0.26 CI 0.44 respectively in Visit 1; Mean 1.55, CI 0.77 and Mean 0.12 CI 0.46 respectively in Visit 2. Movement Wizard: Mean 2.58, CI 0.78 and Mean 0.52 CI 0.5 respectively in Visit 1; Mean 3.97, CI 1.63 and Mean 1.16 CI 1.28 respectively in Visit 2). The Scaffolding group's usage of the Rotation Wizard increased in Visit 2, while the Combination group's usage decreased.

Both groups increased their usage of the Movement Wizard in Visit 2, and used it much more than the Rotation Wizard. The Combination group used more Hints than Wizards in both visits, while the Scaffolding group used Hints more than the Rotation and less than the Movement Wizard (Mean 2.56, CI 0.82 and Mean 2.57 CI 0.79 respectively in Visit 1; Mean 1.55, CI 1.42 and Mean 1.57 CI 1.26 respectively in Visit 2).

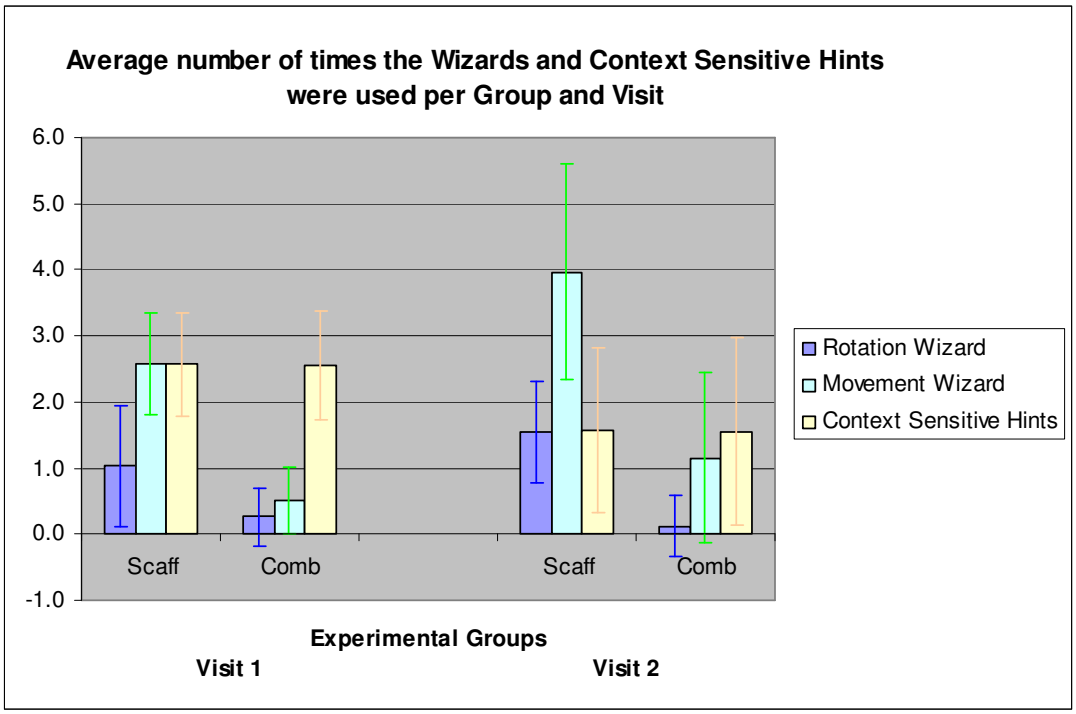


Figure 9.13: Column chart showing usage of Wizards and Total Context Sensitive Hints per Group and Visit. Vertical bars show confidence intervals.

For overall usage of Context Sensitive Hints, the groups were equivalent, with usage dropping in Visit 2. For usage per screen, the Scaffolding group used more Hints than the Combination group in all cases: it used 3D Window Hints much more (Mean 1.2, CI 0.42 and Mean 0.4 CI 0.71 respectively in Visit 1; Mean 0.12, CI 0.68 and Mean 0.02 CI 0.73 respectively in Visit 2), and both groups decreased usage in Visit 2; it used Triggerset Hints more in Visit 1 and the same amount in Visit 2 (its usage decreased, while the Combination group's remained similar) (Mean 0.97, CI 0.81 and Mean 0.62 CI 0.42 respectively in Visit 1; Mean 0.65, CI 0.79 and Mean 0.65 CI 1.18 respectively in Visit 2); and it used Main Menu Hints more in both visits, and increased its usage in Visit 2, while that of the Combination group decreased (Mean 0.4, CI 1.12 and Mean 0.28 CI 1.05 respectively in Visit 1; Mean 0.8, CI 1.18 and Mean 0.16 CI 0.74 respectively in Visit 2).

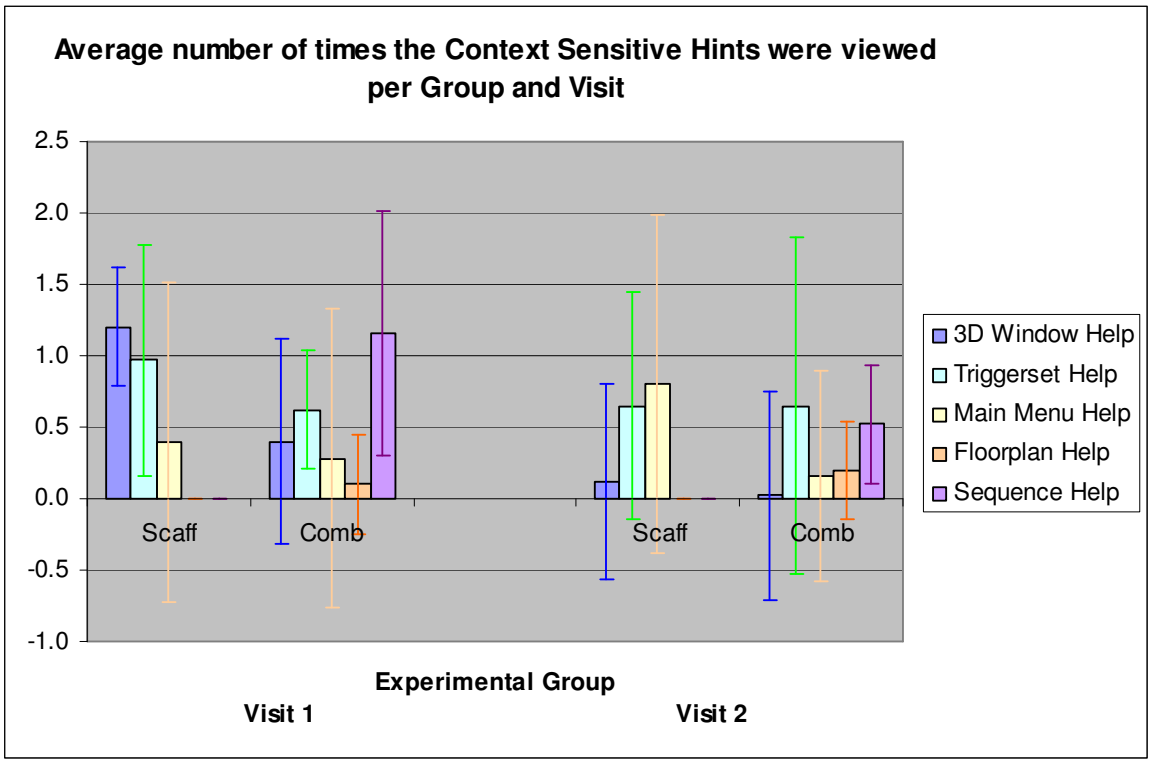


Figure 9.14: Column chart showing usage of Context Sensitive Hints from each screen, per Group and Visit. Vertical bars show confidence intervals.

The Context Sensitive Hints that were only available to the Combination group were the Floorplan and Sequence Hints. The Combination group used the Floorplan Hints very seldom, but it doubled in Visit 2 (Mean 0.1, CI 0.34 in Visit 1 and Mean 0.2, CI 0.34 in Visit 2). They used the Sequence Hints much more often, but it decreased in Visit 2 (Mean 1.16, CI 0.86 in Visit 1 and Mean 0.52, CI 0.42 in Visit 2).

9.7.4. QUIS 7 Questionnaire analysis

We conducted two analyses on the QUIS 7 questionnaire scores: *Total* score; and the *Overall*, *Screen*, *Terminology* and *Learning* components. These variables describe the satisfaction of participants with various aspects of the system. Table G6 in Appendix G displays descriptive statistics for these variables.

QUIS 7 Total

To investigate the effect of representations, scaffolding and repeated experience on the total QUIS 7 score, we conducted a factorial mixed-design ANOVA, with Group as the between-subjects IV, Visit as the repeated measures IV, and QUIS 7 score as the DV. Our analysis revealed a highly significant main effect for both Group ($F(3,129) = 4.1090$, $p < 0.01$) and Visit ($F(1,120) = 85.686$, $p < 0.000001$), but no significant interaction effect ($p = 0.873$). Figure 9.15 displays the means plot for QUIS 7 Total on Group and

Visit, showing the almost-identical score distributions for the two sessions, and the increase of QUIS 7 scores for all groups from Visit 1 to Visit 2.

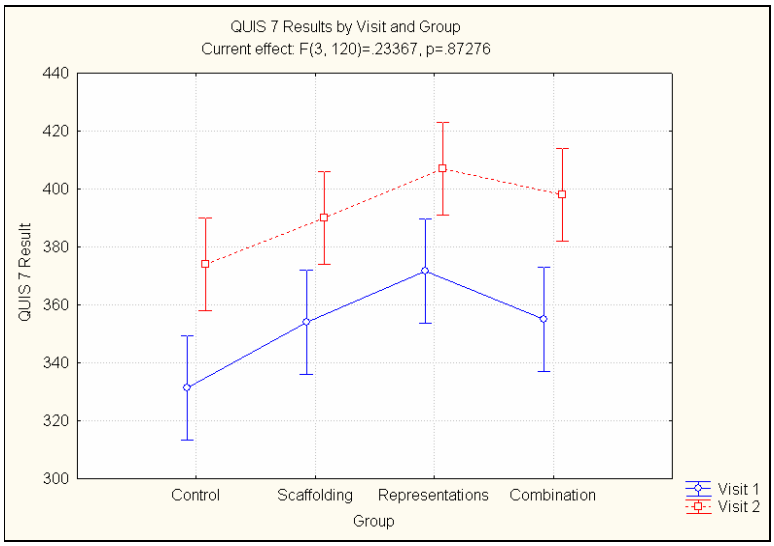


Figure 9.15: Means plot showing the QUIS 7 scores across Group and Visit.

We conducted post-hoc LSD tests to determine the nature of the significant main effect on Group. The Control group scored significantly less than the Representation and Combination groups ($p < 0.001$ and $p < 0.05$ respectively). There were no other significant differences between groups.

QUIS 7 Components

We conducted a factorial mixed-design *multivariate* ANOVA on QUIS 7 components, with Group as the between-subjects IV, Visit as the repeated measures IV, and QUIS Overall, Screen, Terminology and Learning as the DVs. As with the total QUIS 7 scores, our analysis revealed a highly significant main effect for Group ($F(12,309.84) = 2.7277$, $p < 0.01$) and Visit ($F(4,117) = 33.719$, $p < 0.00001$), but no significant interaction effect ($p = 0.1024$). The means plots for the effect of Group and Visit on QUIS 7 components are presented in Figure 9.16.

We conducted post-hoc LSD tests to determine the nature of the significant main effect on Group.

- *QUIS Overall*: the Representation group scored significantly higher than the Control and Scaffolding groups ($p < 0.01$ and $p < 0.05$ respectively).
- *Screen*: the Control group scored highly significantly lower than the Representation and Combination groups ($p < 0.01$ for both).

- *Terminology*: the Representation group scored significantly more than all other groups: $p < 0.0001$ for Control; $p < 0.05$ for Scaffolding and $p < 0.01$ for Combination. The Scaffolding group also scored significantly more than the Control group ($p < 0.05$).
- *Learning*: the Control group scored highly significantly less than the Combination group ($p < 0.01$).

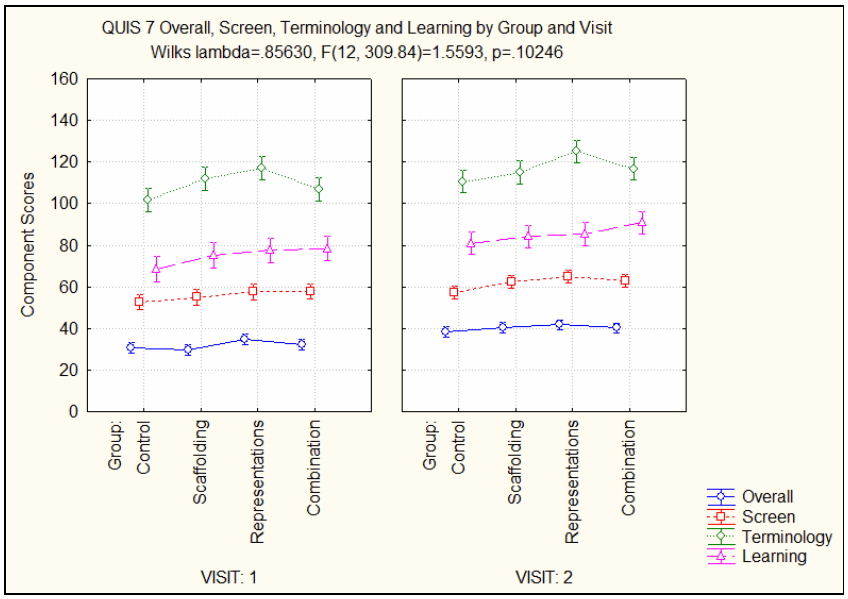


Figure 9.16: Means plots showing the QUIS 7 Component scores across Group and Visit.

We conducted post-hoc LSD tests to determine whether every QUIS 7 component was significantly higher in Visit 2, and found this to be the case. All differences were highly significant at $p < 0.0001$.

9.7.5. System Questionnaire analysis

We conducted two analyses on the System Questionnaire results: the *Total* score; and its *Triggerset* and *Timeline* components. These variables provide the ability to compare satisfaction with the two main VRBridge authoring mechanisms. Table G7 in Appendix G displays descriptive statistics for these variables.

System Total

To investigate the effects of representations, scaffolding and repeated experience on the total System Questionnaire score, we conducted a factorial mixed-design ANOVA, with Group as the between-subjects IV, Visit as the repeated measures IV, and System Questionnaire Total as the DV. Our analysis revealed a highly significant main effect for Visit ($F(1,120) = 65.264$, $p < 0.000001$), but no significant main effect for Group ($p = 0.336$) or interaction ($p = 0.577$). Figure 9.17 shows the means plot for the effect of Group and Visit on System Questionnaire Total. In both visits, the Control and Scaffolding groups score less than the

Representation and Combination groups. Our analysis into the Triggerset and Timeline components reveals opposing results which explain the lack of significance across Group.

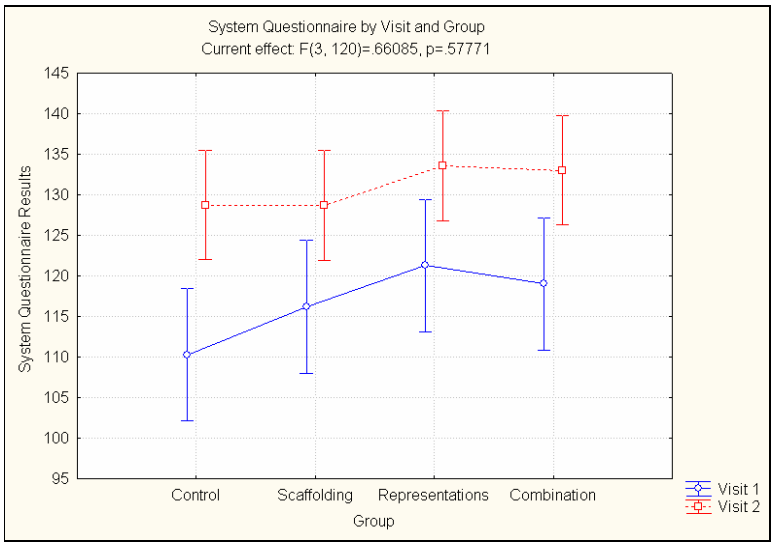


Figure 9.17: Means plots showing the effects of Group and Visit on System Questionnaire Total scores.

System Components

We conducted a factorial mixed-design *multivariate* ANOVA, with Group as the between-subjects IV, Visit as the repeated measures IV, and Triggersets and Timelines as the DVs. We found a significant main effect for Group ($F(6,238) = 2.6769$, $p < 0.05$), a highly significant main effect for Visit ($F(2,119) = 30.575$, $p < 0.00001$), but no significant interaction effect ($p = 0.8607$). The means plots for the effect of Group and Visit on Triggersets and Timelines are presented in Figure 9.18.

The graphs of Triggersets and Timelines show interesting differences in both visits. For Triggersets, the Combination and Representation groups score higher than the other two; but for Timelines this is reversed. We conducted post-hoc LSD tests to understand the extent of the differences.

- *Triggersets*: there are no significant differences across visits, but the Control group scores significantly lower than the Representation group ($p < 0.05$) in Visit 1.
- *Timelines*: the Combination group scores significantly lower than the Control and Scaffolding groups ($p < 0.05$ for both) in both visits.

We conducted post-hoc LSD tests to determine whether the scores for Timelines and Triggersets increased significantly in Visit 2, and found this to be the case ($p < 0.000001$ for both).

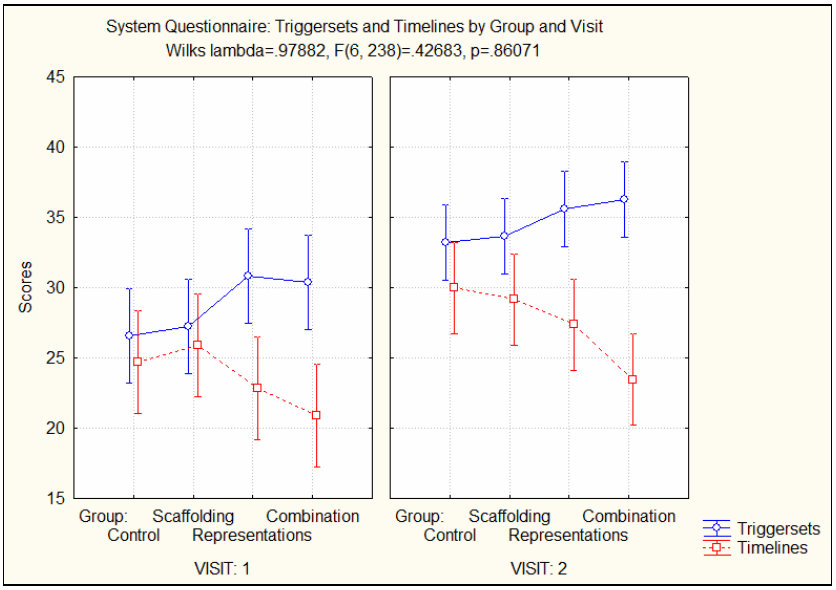


Figure 9.18: Means plots showing the effects of Group and Visit on Triggerset and Timeline scores.

9.7.6. Representations Questionnaire analysis

We conducted two analyses on the Representations Questionnaire scores: the *Total* score; and its *Floorplan*, *Sequence Diagram* and *Run Mode* components. These variables compare satisfaction with each representation. Table G8 in Appendix G displays descriptive statistics for the Representations Questionnaire.

Representations Total

To investigate the effects of the addition of scaffolding to representations, and repeated experience on the Total Representations Questionnaire score, we conducted a factorial mixed-design ANOVA, with Group as the between-subjects IV, Visit as the repeated measures IV, and Representations Questionnaire Total as the DV. Our analysis revealed no significant effect for Group ($p = 0.128$), Visit ($p = 0.44$), or interaction ($p = 0.878$), although the Combination group scores lower than the Representation group in both visits, with the Visit 2 scores for both groups being slightly higher than those for Visit 1.

Representations Components

We conducted a factorial mixed-design *multivariate* ANOVA on Representations Questionnaire components, with Group as the between-subjects IV, Visit as the repeated measures IV, and Floorplan, Sequence and Run Mode as the DVs. Our analysis revealed no significant effect for Group ($p = 0.11037$), Visit ($p = 0.183$), or interaction ($p = 0.11038$). The means plots for the effect of Group and Visit on Floorplan, Sequence and Run Mode are presented in Figure 9.19. The scores for each component show divergences both from each other and across visits. This indicated that there might be significant differences hidden by the combination of

variables. It also indicated a reason for the lack of results for the Representations Questionnaire as a whole. We conducted post-hoc Scheffe tests to determine if there were significant results for each variable.

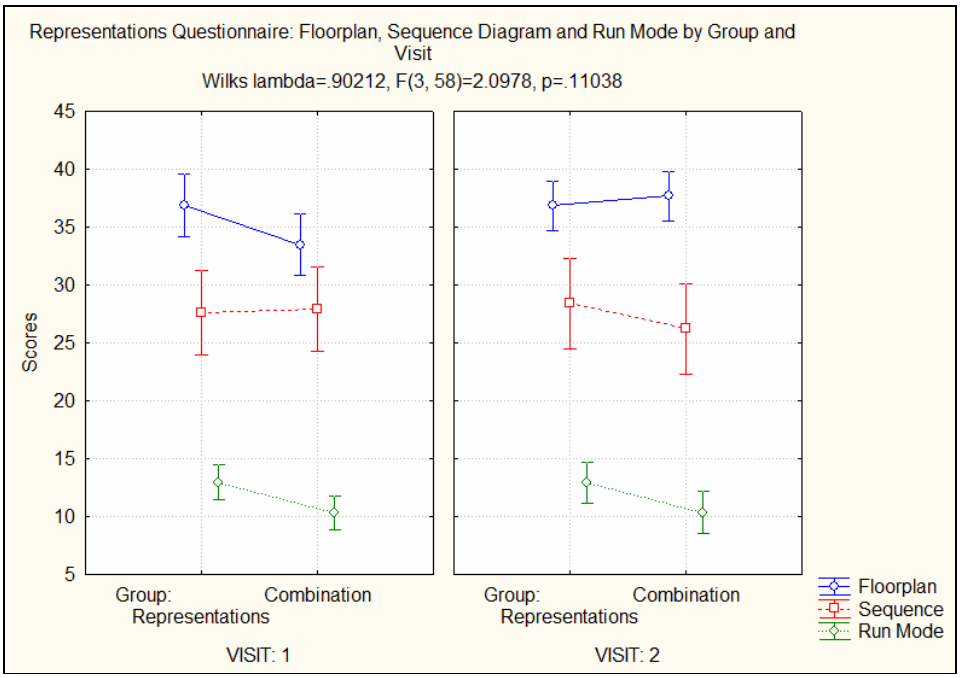


Figure 9.19: Means plots showing the effects of Group and Visit on Floorplan, Sequence and Run Mode scores.

- *Floorplan*: In Visit 1, the difference between the Representation and Combination groups approaches significance, with the Representation group score being higher ($p = 0.0503$). The result for Visit is significant across groups ($p < 0.05$). This is entirely based on the difference between visits for the Combination group, as the score for the Representation group remains similar, while that of the Combination group is significantly higher in Visit 2 ($p < 0.01$).
- *Sequence Diagram*: none of the results is significant; however it is interesting to note that the Sequence pattern is opposite to that of the Floorplan. In Visit 1, the Combination group score is slightly higher than that of the Representation group, while in Visit 2, it is lower.
- *Run Mode*: the score for the Representation group is significantly higher than that for the Combination group ($p < 0.05$). The scores for both groups remain the same across visits.

9.7.7. Scaffolding Questionnaire analysis

We conducted two analyses on the Scaffolding Questionnaire scores: *Total Scaffolding* score; and its *Tooltips*, *Context Sensitive Hints* and *Wizards* components. These variables compare satisfaction with each type of scaffolding. Table G9 in Appendix G displays descriptive statistics for the Scaffolding Questionnaire.

Scaffolding Total

To investigate the effects of the addition of representations to scaffolding, and repeated experience on the Total Scaffolding Questionnaire score, we conducted a factorial mixed-design ANOVA, with Group as the between-subjects IV, Visit as the repeated measures IV, and Scaffolding Questionnaire Total as the DV. Our analysis revealed no significant main effect for Group ($p = 0.9703$) or Visit ($p = 0.178$), but a significant interaction effect ($F(1,53) = 5.4417$, $p < 0.05$). The means plot for this effect is presented in Figure 9.20.

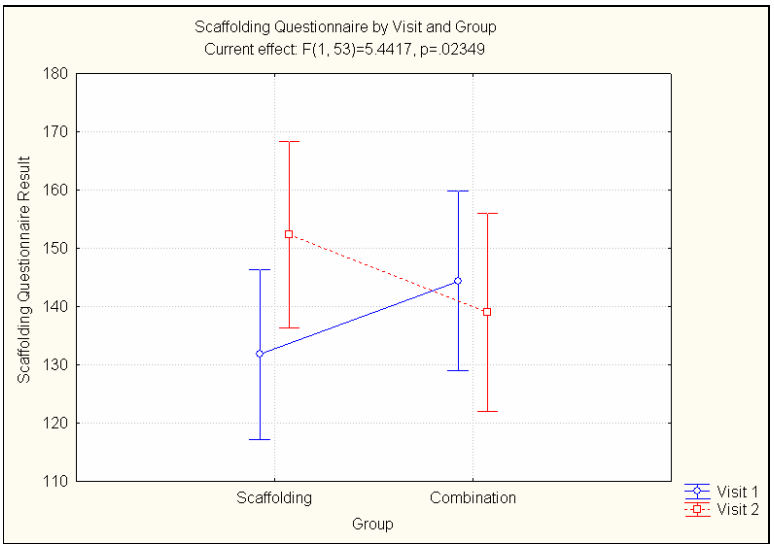


Figure 9.20: Means plot showing the effects of Group and Visit on Scaffolding Questionnaire scores.

In Visit 1 the Combination group score is higher than that of the Scaffolding group; however, in Visit 2, this is reversed. In addition, while the Scaffolding group scores increase across visits, the Combination group scores decrease. We conducted post-hoc Scheffe tests to see if any of these effects was significant. The Scaffolding group's Visit 2 score is significantly higher than its Visit 1 score ($p < 0.01$).

Scaffolding Components

We conducted a factorial mixed-design *multivariate* ANOVA on Scaffolding Questionnaire components, with Group as the between-subjects IV, Visit as the repeated measures IV, and Tooltips, Context Sensitive Hints and Wizards as the DVs. Our analysis revealed no significant effect for Group ($p = 0.0916$) or the interaction ($p = 0.4025$), but a significant main effect for Visit ($F(3,58) = 3.1583$, $p < 0.05$). The means plots for these effects are presented in Figure 9.21.

All of the variables increase at least slightly between visits. However, they differ in their relationships across groups: the Context Sensitive Hint scores are higher for the Combination group; the Tooltips scores are slightly lower for the Combination group; and the Wizards scores are equal. We conducted post-hoc Scheffe

tests to determine whether any of these differences was significant. For *Wizards*, both groups show significantly increased satisfaction in Visit 2 ($p > 0.01$).

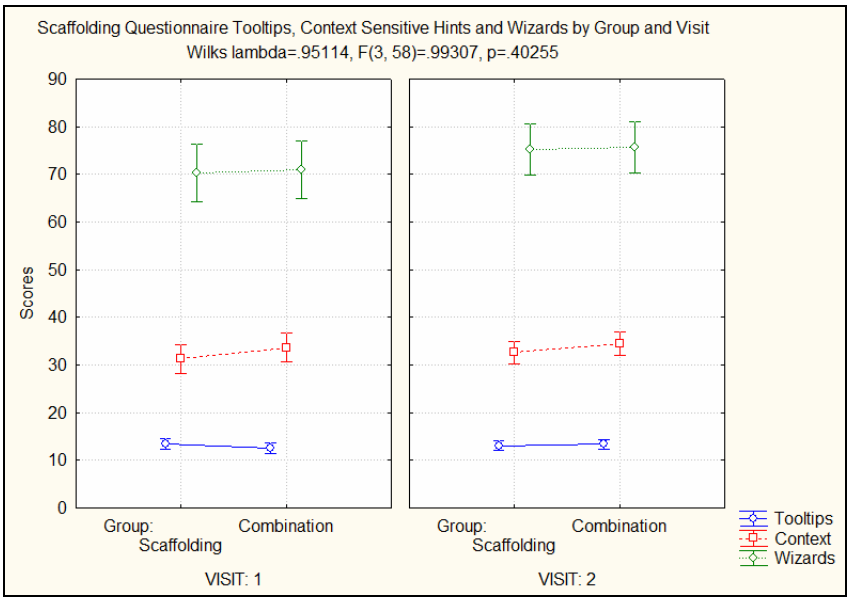


Figure 9.21: Means plots of Tooltip, Context Sensitive Hint and Wizard scores per Group and Visit.

9.8. Discussion

Each measure evaluated a different aspect of participant experiences: *Task* and *XML Explore* scores measured how well participants performed on tasks and explored the VR domain and VRBridge; the logging data measured the extent to which participants used VRBridge; and the questionnaires measured satisfaction with VRBridge and the design aids. These provide a complex and rich description of the relative experiences of participants. The value of providing multiple measures is that we can triangulate them and assess how they interact. E.g., computer logging is compared against Task scores, XML Explore scores and Questionnaire results to gauge how differences in system usage relate to task performance, effective exploration and satisfaction. Table 9.7 shows how our performance criteria relate to the measures, which we use to guide our discussion of the evidence for each of our hypotheses. Tables 9.8 and 9.9 provide summaries by showing the group rankings for all performance measures which compared the four groups.

Performance	**Description**	**Related Measures**
Task Score	Task accuracy and skill	XML Task Scores
Effective Exploration	VRBridge: General expertise	XML Explore Total; Logs of 3D Pause, Run Mode, Context Sensitive Hints, Tooltips

Performance	Description	Related Measures
	VRBridge: Customise Triggersets	XML Variable Sequencing; Logs of Triggersets, Sequence Diagram
	VRBridge: Synchronise Timelines	XML Inevitable Sequencing; Logs of Timelines
	VRBridge: Customise space	XML Features; Logs of waypoints and locations
	VRBridge: Customise objects	XML Features; Logs of Object Panel, Floorplan
	Dynamic VR: Player freedom	XML Player Freedom; Logs of Sequence Diagram
	Dynamic VR: Exploit space	XML Creativity; Logs of waypoints, locations, Floorplan
	Dynamic VR: Interesting narrative	XML Creativity; Logs of previews
	Dynamic VR: Natural interactions	XML Creativity; Logs of wizards
Satisfaction	With VRBridge and design aids	Questionnaires
Learning	Improved Task, Exploration and Satisfaction in Visit 2	Changes in all measures

Table 9.7: Table of performance criteria, showing how they relate to the measures.

Measure	Visit 1				Visit 2			
	Rank 1	Rank 2	Rank 3	Rank 4	Rank 1	Rank 2	Rank 3	Rank 4
TrigAuth	Repres	Comb	Scaff	Control	Repres	Scaff	Comb	Control
TimeCreate	Scaff	Control	Comb	Repres	Scaff	Control	Repres	Comb
TimeAuth	Scaff	Control	Comb	Repres	Scaff	Control	Comb	Repres
ObjPnlMan	Repres	Comb	Scaff	Control	Comb	Scaff	Repres	Control
WayCreate LocCreate	Comb	Repres	Scaff	Control	Comb	Repres	Control	Scaff
AnimPrev	Scaff	Control	Repres	Comb	Control	Scaff	Repres	Comb
SndPrev	Scaff	Comb	Control	Repres	Scaff	Comb	Control	Repres
TrigMan	Repres	Control	Comb	Scaff	Repres	Comb	Control	Scaff
Pause3D	Repres	Scaff	Comb	Control	Repres	Comb	Scaff	Control
Loc/Way3D	Scaff	Control	Repres	Comb	Scaff	Control	Comb	Repres
QUIS7 Tot	Repres*	Comb^	Scaff	Cont*^	Repres*	Comb^	Scaff	Cont*^
QUISOveral	Rep*^	Comb	Cont*	Scaff^	Repres	Comb	Scaff	Control
QUISScreen	Repres*	Comb^	Scaff	Cont*^	Repres*	Comb^	Scaff	Cont*^
QUISTerm	Rep*^&	Scaff*#	Comb^	Cont&#	Rep*^&	Scaff*#	Comb^	Cont&#

Measure	Visit 1				Visit 2			
	Rank 1	**Rank 2**	**Rank 3**	**Rank 4**	**Rank 1**	**Rank 2**	**Rank 3**	**Rank 4**
QUISLearn	Comb*	Repres	Scaff	Control*	Comb*	Repres	Scaff	Control*
SysQuesTot	Repres	Comb	Scaff	Control	Repres	Comb	Scaff	Control
SysTrig	Repres*	Comb	Scaff	Control*	Comb	Repres	Scaff	Control
SysTime	Scaff*	Control^	Repres	Comb*^	Control*	Scaff^	Repres	Comb*^

Table 9.8: Table of group ranking on performance measures for Visit 1 and Visit 2. The *^ and # symbols denote significant differences.

	Task 1				Task 2				Task 3			
	1	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**
Task Score	Rep*	Scaf^	Com#	Cont *^#	Rep*	Com ^	Scaf#	Cont *^#	Rep*	Com^	Scaf#	Cont *^#
XML Expl	Scaf	Com	Rep	Cont	Scaf*	Rep^	Com	Cont *^	Rep*	Scaf	Com	Cont*
Var Seq	Com *	Scaf^	Rep#	Cont *^#	Rep#	Com *	Scaf^	Cont *^#	Rep#	Com*	Scaf^	Cont *^#
Inev Seq	Scaf*	Cont	Rep	Com *	Scaf*	Cont	Rep	Com *	Scaf*	Rep	Cont	Com*
Creat	Scaf*	Com	Rep^	Cont *^	Rep^	Scaf*	Com	Cont *^	Scaf*	Rep^	Com	Cont*^
Feat	Com *^	Rep# &	Scaf*#	Cont ^&	Rep# &	Com *^	Scaf* #	Cont ^&	Rep# &	Com*^	Scaf* #	Cont^ &
User Free	Com *	Rep^	Scaf#	Cont *^#	Com *	Scaf#	Rep^	Cont *^#	Rep^	Scaf#	Com *	Cont*^ #

Table 9.9: Table of group ranking on performance measures for Task 1, 2 and 3. The *^& and # symbols denote significant differences.

9.8.1. Hypothesis 1

Both representations and scaffolding will improve participant performance. Representations will improve task performance, effective exploration and satisfaction more and scaffolding will improve learning more. The combination will improve performance most.

Task Completion

The *Task Scores* metric provides information on task completion. Our results show that the Control group performs significantly worse than all other groups on every task, even Task 1 where the experimenter provided assistance. **Therefore, both representations and scaffolding improve participant task performance**. However, there are no significant differences between the other groups. The Representation group scores the highest in Task 1, the Representation and Combination groups score highest in Task 2, and all three score the same in Task 3. This shows that those with representations tend to perform the task better initially, but there are no statistically significant differences except with the Control group. **Therefore, the combination of representations and scaffolding does not improve task performance over either individually**.

Effective Exploration

For effective exploration, we examine general expertise, expertise with VRBridge and expertise in the domain. Both the *XML Explore* scores and the logging results provide evidence here.

For *general expertise*, we examine the *XML Explore Total* and usage of *3D pausing*, as this provides finer control over design by allowing participants to step through their VE. For *XML Explore Total*, only the Scaffolding group scores significantly higher than the Control group. However, if each task is examined separately, the Representation group scores significantly higher than the Control group in both Task 2 and 3, where the Scaffolding group is only significantly higher in Task 2. The Control group does not score significantly worse than other groups in Task 1. The Representation group improves the most across tasks, going from almost equal to the Control group in Task 1 to first in Task 3. The Combination group scores higher than the Control group in all three tasks and highest in Task 1, but scores lower than the Representation and Scaffolding groups in Tasks 2 and 3. The Control group *pauses* the least in both visits, while the Representation group pauses the most. The Combination group pauses less than the Scaffolding group in Visit 1, but more in Visit 2.

For *general expertise* variables that only apply to representations or scaffolding, we examine *Run Mode starting and viewing*, usage of *Context Sensitive Hints* and *Tooltips*. In Visit 2, the Combination group starts the *Run Mode* much less than the Representation group, but views it longer (in Visit 1, the Representation group's viewing time is longer). This difference between starting and viewing times suggests that the Combination group spends longer viewing work dynamically alongside the representations in Visit 2, which also connects with its longer viewing times for the representations. However, this pattern combined with a lack of *3D pausing* suggests that the Representation group uses the *Run Mode* with more discrimination. For *Context Sensitive Hints*, the Scaffolding group uses far more common hints than the Combination group. Only the Combination group manipulates the *Tooltips*.

For *customising Triggersets*, the Control group scores significantly lower than all other groups on *Variable Sequencing*; the Representation and Combination groups score highest. The Control group authors fewest *Triggersets*, but does more *Triggerset manipulations* than the Scaffolding and Combination groups in Visit 1, and more than the Scaffolding group in Visit 2. However, it is the only group not to increase its manipulations between visits. The Representation group authors and manipulates the most *Triggersets*. The Combination group does fewer *Sequence Diagram* manipulations than the Representation group, while viewing longer. This suggests that it may leave the diagram open without exploring it, as manipulations show more focused interaction with the diagram.

For *synchronising Timelines*, the Scaffolding and Control groups score highest in *Inevitable Sequencing*, while the Combination group scores lowest, significantly lower than the Scaffolding group. The Control group creates and authors more *Timelines* than the Representation and Combination groups, but fewer than the Scaffolding group.

For *customising space and objects*, we examine *XML Features*, and logging of *waypoints*, *locations* and the *Object Panel*. The Control and Scaffolding groups score significantly lower than the Representation and Combination groups on *XML Features*. The Control group does the fewest *object panel manipulations*. It and the Scaffolding group create the fewest *locations* and *waypoints*. The Representations and Combination groups create the most *waypoints* and *locations*. They do the most *object panel manipulations* in Visit 1, but the Combination and Scaffolding groups do the most in Visit 2. The Control group *views waypoints and locations in the 3D world* less than the Scaffolding group but more than the Representation and Combination groups. In Visit 1, the Combination group opens the *Floorplan* more often and views it longer than the Representation group, but does fewer manipulations. In Visit 2, it uses the *Floorplan* more in all respects.

For *exploiting space, interesting narrative and natural interactions*, we examine *XML Creativity*. The Control scores the lowest, significantly lower than the Representation and Scaffolding groups. The Scaffolding and Combination groups score highest in Task 1, but the Representation group scores highest in Task 2. The Combination group scores almost as low as the Control group in Tasks 2 and 3. For logging, we can examine usage of previews for narrative interest. The Control and Scaffolding groups view the most *animation previews*, while the Combination group views the fewest in both visits. For *sound previews*, the Representation group views the least, and the Scaffolding and Combination groups view the most. For scaffolding usage, the Scaffolding group uses both Wizards more than the Combination group. The logging results combined with *XML Features* suggest that the Representation and Combination groups explore the VE space more effectively, but that the Scaffolding group is equivalent in other forms of *domain expertise*.

For *promoting player freedom*, we examine *XML Player Freedom*, where the Control group scores significantly less than all other groups, which have no significant differences. We now combine these results for overall evidence about *effective exploration*. For Task 1, the Control group scores less than the other groups but does not explore significantly less well. However, even in Visit 1, it explores the domain less effectively, as shown by the *Variable Sequencing*, *Player Freedom* and *Creativity* scores. In particular, the first two capture the indeterminism of VR interactions and therefore convey important domain-specific expertise, and ability to work effectively with Triggersets. The only component in which it scores better than others is *Inevitable Sequencing*, which matches the *Timeline* logging data. This is the weakest component of the *XML Explore* score in domain expertise terms, as it conveys skill at creating interactions that will not change according to player interaction. However, it shows that the Control group does author interactions successfully with Timelines. For Tasks 2 and 3, the Control group scores significantly less than the other groups on both task and exploration, and generally performs fewer actions.

The usage of *previews* and *viewing waypoints and locations in 3D* shows that the Control group is interested in understanding the space and content. However, it and the Scaffolding group's larger usage are more likely a result of the availability of the Floorplan and other visual resources to the Representation and Combination groups, than a result of greater interest, especially when compared with the *Creativity* scores. Similarly, the success of the Representation and Combination groups at *creating locations and waypoints* is probably because they can use direct manipulation on the Floorplan. They also have more help in working with *Triggersets* from the *Sequence Diagram* and *Run Mode*. Therefore, they are more likely to use Triggersets over Timelines because of the additional assistance. The fact that the Scaffolding and Control groups create the most *Timelines* and score highest on *Inevitable Sequencing* supports this theory. Another support comes from the increased viewing of *3D Window* and *Triggerset Hints* by the Scaffolding group, as the Combination group had access to other sources of help: the Floorplan and Sequence Diagram.

The evidence leads us to the conclusion that **the Control group explores less effectively than other groups** and **the Representation group in general explores more effectively than the Scaffolding group**. The second conclusion is not unequivocal because of the lack of significant results. However, the combination of the following factors makes it highly likely: the Representation group's increase in *XML Explore* to become the top scorer; its top scores in *Variable Sequencing* and *Player Freedom*; its high usage of important features for effective exploration (e.g., 3D pause, Triggerset and object panel usage); and its similarity to the Combination group on most of these items.

Evidence for the effectiveness of the Combination group's exploration is also mixed. It scores significantly lowest on *Inevitable Sequencing*. However, it scores as well as the Representation group on *XML Features* and *Variable Sequencing* and as well as the Representation and Scaffolding groups on *Player Freedom*. It

also performs a relatively high number of *object panel* and *Triggerset manipulations*, and *waypoint and location creation*. In Task 1, it scores highest on *Variable Sequencing*, *Features* and *Player Freedom* (and second highest on *Creativity*). However, the overall *XML Creativity* score shows that it is less successful at domain-level expertise in terms of creating an interesting narrative and natural interactions than the Scaffolding and Representation groups. It also explores the scaffolding less than the Scaffolding group and the representations less effectively than the Representation group. While in some ways the Combination group explores more effectively than the Scaffolding group, overall it performs worse than the Representations and no better than the Scaffolding group.

Therefore, the representations and scaffolding both encourage more effective exploration, but the combination does not improve exploration over either individually. In fact, adding scaffolding to representations decreases effective exploration.

Satisfaction

The Control group scores lower than all others on the *Quis 7 Questionnaire* results. For *Quis 7 Total* and *Screen*, the Representation and Combination groups score significantly more than the Control group. For *Overall* satisfaction, the Representation group scores significantly more than the Control and Scaffolding groups. For *Terminology* satisfaction, both Representation and Scaffolding groups score significantly more than the Control group; however, the Representation group also scores significantly more than the Scaffolding and Combination groups. This is the only QUIS 7 measure where the Scaffolding group scores significantly more than the Control group, which makes sense, as the Scaffolding directly addresses understanding of terminology. Interestingly, the Combination group scores lower than the Scaffolding group in Visit 1, but higher in Visit 2. For *Learning* satisfaction, the Combination and Representation groups score the highest, and the Combination group scores significantly more than the Control group.

The results are similar for the *System Questionnaire*, although they are not significant. The Control group scores lowest, and the Representation and Combination groups score the highest. For *Triggerset* satisfaction, the Control group scores lower than all groups, significantly lower than the Representation group in Visit 1. However, for *Timeline* satisfaction, the Control and Scaffolding groups score highest, significantly higher than the Combination group. This *Timeline* result explains the overall lack of significance in the System result (as satisfaction with the components of *Timelines* and *Triggersets* is opposite) and connects to the success of the Control group at *Inevitable Sequencing*. It is worth noting that the difference in *Triggerset* satisfaction decreases across visits, as the Control group becomes more satisfied relative to the other groups.

Therefore, the **Control group is much less satisfied than the Representation and Combination groups, and marginally less satisfied than the Scaffolding group**, and **the Representation group is more satisfied than the Scaffolding group**.

For the *Representation questionnaire*, the Combination group scores lower overall than the Representation group, but not significantly. For *Floorplan* satisfaction, its score is lower than that for the Representation group in Visit 1, but higher in Visit 2, and is significantly higher than its own result in Visit 1. For *Sequence Diagram* satisfaction, its score is higher than that for the Representation group in Visit 1 and lower in Visit 2. For *Run Mode* satisfaction, it scores significantly lower than the Representation group, which connects with its lack of successful Run Mode usage. For the *Scaffolding questionnaire*, there are no significant difference between the groups, but the Combination group score drops and the Scaffolding group score increases across visits. For *Context Hint* satisfaction, the Combination score is higher than that for Scaffolding in both visits, and for *Tooltip* satisfaction it changes from lower to similar. Both groups feel the same *Wizard* satisfaction. Therefore, the Combination group experiences slightly higher overall satisfaction than the Scaffolding group, but slightly lower than the Representation group. However, the result for *Learning* satisfaction, and the improved results for the *Floorplan* and *Scaffolding* satisfaction show that it feels more satisfaction with learning than any other group.

Therefore, representations and scaffolding both improve satisfaction. The combination improves satisfaction with learning more than either alone. However, in general, the addition of scaffolding to representations decreases satisfaction.

Learning

Although all groups increase their *XML Explore* and questionnaire scores across tasks and visits, the Control group is the only group not to increase *Triggerset manipulations* between visits. The difference between the effectiveness of its exploration and that of other groups also increases to become significant after Task 1, which suggests that the other groups learn more during Task 1 and so are able to accomplish the other tasks more effectively. Therefore, **the Control group learns less than the other groups**.

Next we examine evidence for whether the Scaffolding group learns better than the Representation group. They perform similarly on task completion overall. The Scaffolding and Combination groups improve more than the Representation group between Tasks 2 and 3, as they move from second and third position to equivalent with the Representation group. However, the Representation group's score drops the least between Tasks 1 and 2, when the experimenter assistance is removed. For *XML Explore*, the Scaffolding group scores highest in Tasks 1 and 2. However, the Representation group improves more with each task, to become the top scorer in Task 3. If we examine logging results, we find that the Scaffolding group is only

one not to decrease its *object panel manipulations* in Visit 2, giving it and the Combination group the highest usage. However, there are no other variables where it increases its usage and others do not. Therefore, **we cannot assert that the Scaffolding group learns more effectively than the Representation group**.

There is some evidence that the Combination group learns better than the Representation and Scaffolding groups. It increases its exploration in Visit 2 more than other groups, e.g., *Triggerset manipulations*, *Timeline creation* and *manipulation*, *animation* and *sound previews*, *Run Mode viewing*, *Movement Wizard*, *Triggerset Hints* and *Floorplan Hints.* It also increases its position compared to other groups in some usage, e.g., *3D pause*, *location creation*, *viewing locations and waypoints*, *Floorplan manipulations* and *Triggerset Hints*. More exploration must be coupled with increased success for evidence that learning has occurred. If we examine *task scores* again, we see that the Combination group improves its position against other groups, beginning with Task 1 as the third highest group, and moving up to become the second highest group, and then almost equal to Representations in Task 3. However, it does not *explore* more effectively than the Representation and Scaffolding groups; its increase in *XML Explore* from Task 1 to Task 2 is less than that of the Representation and Scaffolding groups, and its improvement from Task 2 to Task 3 is less than that of the Control and Representation groups. Therefore, the evidence for improved learning by the Combination group is not unqualified. **Although there is some evidence for improved learning by the Combination group, there is also evidence against it. Therefore, we cannot state that the combination of scaffolding and representations improves learning over either individually.**

The addition of scaffolding seems most to improve initial exploration of VRBridge, as the Combination and Scaffolding groups have the top scores for *Creativity* and *Variable Sequencing*, the Scaffolding group has the top score for *Inevitable Sequencing* and the Combination group has the top score for *Features* and *Player Freedom* in Task 1. By Task 2, the Scaffolding group only has the top score in *Inevitable Sequencing*, and the Representation group has become the top scorer in *Variable Sequencing*, *Creativity*, and *Features*. In addition, the initial support of scaffolding for exploration does not translate to *task scores*, as the Representation group scores highest in Task 1.

Summary

The evidence indicates that we can reject the null hypothesis for Hypothesis 1 for the most part. **All groups perform better than the Control group**, in terms of task accuracy and skilful completion, effective exploration, learning and satisfaction. **The Representation group explores more effectively than the Scaffolding group and is more satisfied with the experience;** however, although it initially shows more expertise in task completion, overall **it does not complete the task more skilfully**. In addition, **the Scaffolding group does not learn better than the Representation group**. **The Combination group performs better than the Control group. However, it does not perform better than the Representation**

and Scaffolding groups on any aspect. **In fact, it performs worse in general than the Representation group**.

The findings suggest that the Combination group has some of the benefits that representations give the Representation group and scaffolding gives the Scaffolding group. However, these do not add up to make it perform better overall. It often performs worse than the Representation group where that group performs less well, and similarly for the Scaffolding group. The combination of representations and scaffolding causes conflict, which lessens the effectiveness of the Combination group. This could be because Combination participants must learn more functionality than any other group, and are therefore more overwhelmed by the new experience. They have more information on screen, demanding their attention. This explanation is supported by the fact that the only participants to turn *Tooltips* off were from the Combination group. As *Tooltips* appear constantly on the interface, turning them off would lessen the amount of information appearing on screen. Another supporting finding is that the difficulties of the Combination group seem to lessen in Visit 2: they show increased satisfaction with the system and are most satisfied with learning. However, their exploration of the system worsens, as does their creativity.

9.8.2. Hypothesis 2

All groups will complete the task more accurately and skilfully in the second session and be more satisfied with the experience.

Task Completion

All groups completed Task 3 more accurately and skilfully than Task 2. Task 1 was a learning task with experimenter assistance and all groups performed best on this task. This result is not conclusive as the tasks were different to each other and their relative difficulty is a factor. However, we can examine the *XML Explore* scores to gain more conclusive results. For *XML Explore Total* and *Creativity*, each task was explored significantly more successfully than the previous one; for *Variable Sequencing* and *Inevitable Sequencing*, the scores for Task 3 are significantly higher than those for Tasks 1 and 2; and for *Features*, Task 2 is significantly higher than Tasks 1 and 3, and Task 3 is significantly higher than Task 1. *Features* is more tied into the task requirements than other measures and it may be that Task 2 required more obvious usage. *Player Freedom* is the only component that does not change between tasks. Therefore, we can state from the evidence that **all groups complete the task more accurately and skilfully in Visit 2**.

Satisfaction

For the *Quis 7 questionnaire*, *System questionnaire* and their components all groups score significantly higher in Visit 2 than in Visit 1. The *Representation* and *Scaffolding questionnaires* are less clear: both have higher scores in Visit 2 than in Visit 1, but this is not significant for either (except for Scaffolding group

satisfaction). Satisfaction with the *Floorplan* is significantly higher in Visit 2, but satisfaction with the *Sequence Diagram* is not, and satisfaction with the *Run Mode* remains the same. Satisfaction with *Tooltips* and *Context Sensitive Hints* remains the same, and the satisfaction with *Wizards* increases significantly. The evidence indicates that we can reject the null hypothesis for Hypothesis 2. **All groups complete tasks more skilfully and accurately, and generally experience more satisfaction during Visit 2**.

9.8.3. Hypothesis 3 and evidence about scaffolding usage

Participants will use the scaffolding less in the second session.

To investigate how scaffolding usage changed across visits, we begin by exploring participant satisfaction. The Scaffolding group was significantly more satisfied with the scaffolding in Visit 2, while the Combination group was slightly less satisfied. The Scaffolding group was slightly less satisfied with the *Tooltips*, while the Combination group was slightly more satisfied. Neither group changed its feelings about the *Context Sensitive Hints*, but both groups were significantly more satisfied with the *Wizards*.

Next we examine the logging data for scaffolding usage over time. Only two participants turned *Tooltips* off. Therefore, most participants experienced them during the entire experiment. The Scaffolding group increased its usage of the *Rotation and Movement Wizards* in Visit 2, while the Combination group decreased its usage of the *Rotation Wizard* and increased its usage of the *Movement Wizard*. Both groups used the *Context Sensitive Hints* less often in Visit 2, although if we examine individual hints the Scaffolding group increased usage of *Main Menu* help and the Combination group increased usage of the *Floorplan* help. This connects with the Combination group's increased usage of the Floorplan in Visit 2. The Combination group used the *Context Sensitive Hints* more than the *Wizards*, while the Scaffolding group used it more than the *Rotation Wizard* but less than the *Movement Wizard*. In general, the Combination group used the *Wizards* much less than the Scaffolding group; this is explained by the fact that the Wizards provide floorplans and graphical help, to which the Combination group would already have access

These results show that we cannot completely reject the null hypothesis for Hypothesis 3. **Although both groups decrease their usage of some of the scaffolding, they also increase some usage**. Participants will fade the scaffolding they do not use, and become more satisfied with the scaffolding that they keep. It was unexpected that so few people turned off the *Tooltips* as they are the most invasive form of scaffolding (compared to *Wizards*, which form part of the task, and *Hints*, which are opened by choice). The fact that the experiment entailed working with VRBridge for only a short time must have influenced this result. It is likely that more scaffolding would be faded over a longer period of time, as users become more familiar with VRBridge.

The most basic forms of scaffolding, which only provide information, are *Context Sensitive Hints* and *Tooltips*. The Hints were generally faded; while the Tooltips were probably ignored (which is also fading, but less measurable). The Wizards provide sophisticated forms of scaffolding and are more integrated with the work. They make initial attempts at authoring easier, and their output can be customised. They may never be faded as they provide output that can be customised. As all groups used Timelines more effectively in Task 3, we can say that they were customised better, but the Wizards were still being used. The danger to development of domain expertise is that the Wizards output Timelines and so might negatively impact effective usage of Triggersets. However, all groups explored Triggersets more effectively in Visit 2. Therefore, while the Wizards were not faded, the consequences are not harmful.

9.8.4. Hypothesis 4 and evidence about representation usage

The Floorplan will be used more and provide more satisfaction than the Sequence Diagram.

We begin by exploring participant satisfaction with the Representations. Both groups were slightly more satisfied with the Representations in Visit 2. The Combination group began less satisfied with the *Floorplan* and increased its satisfaction significantly in Visit 2, while that of the Representation group remained the same. The Combination group began slightly more satisfied with the *Sequence Diagram* and became slightly less satisfied, while the Representation group became slightly more satisfied. The Representation group was significantly more satisfied with the *Run Mode* in both visits.

Next we examine the logging data for information on how representation usage changed over time. Both groups opened and manipulated the *Floorplan* less in Visit 2, but viewed it longer. This suggests that participants increasingly used the Floorplan for reference while they were authoring interactions. Both groups opened, manipulated and viewed the *Sequence Diagram* less in Visit 2. This suggests that it is more useful for initial domain investigations. The same is true of the *Run Mode*, which both groups started less often in Visit 2. The Combination group viewed it longer in Visit 2, possibly showing that they found it increasingly useful, however their satisfaction with it did not increase.

Both groups used the *Floorplan* much more often than the *Sequence Diagram*, and the differences between groups are less with the *Floorplan*. It was also the only representation to be used at least once (by all Representation group members) in Visit 1. Both groups manipulated the *Floorplan* more and viewed it longer than the *Sequence Diagram*, and their viewing times increased in Visit 2 as opposed to decreasing. The satisfaction scores for the Floorplan were also consistently higher than those for the Sequence Diagram. These results show that we can reject the null hypothesis for Hypothesis 4 and state that **the Floorplan is used more and provides more satisfaction than the Sequence Diagram**.

9.9. Synthesis

In this chapter, we have described an experiment to investigate usage of VRBridge, representations and scaffolding in authoring VR interactions. We developed three measures for this purpose: task and exploration scores to measure task performance, questionnaires for satisfaction, and computer logging for participant activity. We developed three tasks for participants to perform over two sessions with VRBridge.

We expected that both representations and scaffolding would improve participant performance, in terms of skilful task completion, effective exploration, satisfaction, and improved learning (Hypothesis 1). We found this to be the case for all measurements, however were unable to support our expectations about the combined effect of representations and scaffolding, and the relative contribution of each. We expected that the combination would improve participant performance more than either alone; however, it only improved on the other groups in satisfaction with learning. We expected that representations would improve skilful task completion, effective exploration and satisfaction more than scaffolding; however, this was only shown for exploration and satisfaction. We also expected that scaffolding would improve learning more than representations, which was not the case.

We expected that all groups would complete tasks more skilfully and accurately, and experience more satisfaction during Visit 2 (Hypothesis 2) and found this to be the case.

We also investigated usage of scaffolding and representations. We expected that participants would fade scaffolding in Visit 2 (Hypothesis 3). We found that only the less sophisticated, non-automatic scaffolding was faded. We expected that the Floorplan would be used and enjoyed more than the Sequence Diagram (Hypothesis 4), and found this to be the case.

We have shown that VRBridge, combined with the design aids, supports more expert tool usage and domain-level authoring. In the next chapter, we discuss our results more deeply, and the further implications of this research, in terms of our larger aims of supporting end-user authoring and creating effective interaction authoring systems.

10. Reflections on VR Authoring, Interaction Design and End-User Creativity

In this chapter, we discuss the implications of our research in the light of our evaluations. We examine evidence for the effectiveness of constructivism in guiding and structuring a design process, specifically in supporting VR interaction design for end-users. To this end, we examine the usefulness of scaffolding and representations, and the contributions of each element. Then we consider the usefulness of VRBridge and impact of the underlying constructivist principles. Finally, we reflect on the broader implications of our research for end-user authoring and designing interactive systems.

10.1. Supporting Effective VR Interaction Authoring in End-Users

To support effective VR interaction authoring by end-users, we developed an authoring tool, VRBridge, and design aids in the form of scaffolding and representations. The design aids provide support for authoring, both in terms of using VRBridge effectively, and developing expertise in VR authoring.

We begin by reviewing our mechanisms for evaluating success. For end-users to author creative products successfully, they must be able to retain interest in the activity while learning and practising skills. They must be able to create products which are sophisticated enough to demonstrate proficiency in the domain and motivate continued work. These points can be organised into two axes along which we define success, described below and depicted graphically in Figure 10.1:

- *Usable Tool*: Provision of a tool that is easy and fun to use, has a gentle learning curve and the power to express interactions commonly found in VEs.
- *Domain Expertise*: The extent to which we have helped the VR design novice to understand the domain conceptually, including its particular challenges and requirements.

If we can show that we have succeeded in these aims, then we have created a product which supports effective VR interaction authoring by end-users. Section 6.3.2 described how we ensured the expressive power of our authoring interface, by designing Triggerset components which could recreate the interactions in a variety of VEs and 3D games created using traditional programming methods. We were able to recreate most interactions, apart from some aspects requiring advanced mechanisms that were unnecessary for our purpose (the creation of a proof-of-concept prototype), e.g., the ability to pick an object up and associate it with another; and the provision of a heads-up display are not supported. Having confirmed that we could recreate common interactions, we needed to show that novices could use VRBridge effectively, both for authoring interactions and learning significant domain characteristics.

The combination of our evaluation studies provides the bulk of our evidence. In the first study (Chapter 7), we used observation, interviews and task success to evaluate how effectively end-users analysed and debugged interactions created with VRBridge. This provided evidence for end-user conceptual understanding of VR interactions, VRBridge ease-of-use, and the effectiveness of our design representations. In the second study (Chapter 9), we used a task performance metric, questionnaires and computer logging to evaluate how skilfully end-users created interactions and explored VRBridge and the domain, how much satisfaction they experienced, and how well they learned over time.

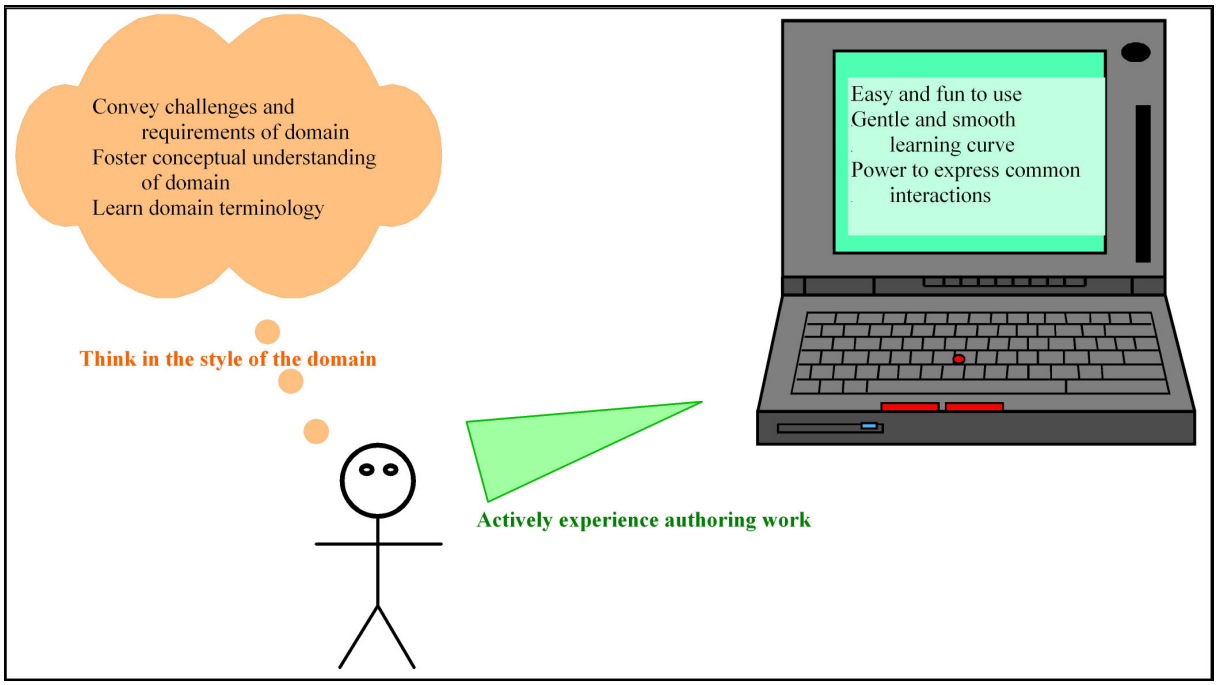


Figure 10.1: Requirements for successful provision of authoring support to end-users.

The components of our task performance metric provide valuable insights into how successfully participants worked with VRBridge and understood the domain. This relates directly to our interest in helping non-expert users design interesting and effective VR, and removing technical barriers. Table 10.1 shows how the task performance metric items relate to our requirements.

Metric Item	Description	Domain or Technical Expertise Shown
Task Completion	Completes task requirements skilfully	Understanding of technical constraints; ability to create typical interactions accurately.
Explore Total	Overall exploration of system and domain	Effective exploration

Metric Item	Description	Domain or Technical Expertise Shown
Variable Sequencing	Creation of divergent and nuanced interactions with Triggersets	Creation of non-linear interactions; Usage of building blocks to create complex and interesting higher-level interactions.
Inevitable Sequencing	Creation of multiple synchronised objects and actions on Timelines, and integration with Triggersets	Sequencing of objects and actions to create synchronised interactions. Indicates more linear interaction programming as interactions are inevitable once started.
Player Freedom	Player involvement in interactions and ability to influence events	Creation of highly interactive environment, and interactions that are reactive to player; awareness of uncertainty of player interaction.
Creativity	Creation of narratively interesting design, natural interactions and exploitation of space.	Awareness of importance of action detail (e.g. turning to face) for effective interactions; ability to create interesting narratives with interactions; promotion of player navigation. Shows the interest in authoring necessary for continued exploration of VRBridge and the domain.
Features	Successful usage of VRBridge features in creating interactions	Technical understanding of utility of VRBridge features for authoring effective interactions.

Table 10.1: Description of task performance metric, showing how items relate to domain and technical expertise.

In particular, *Variable Sequencing* and *Player Freedom* address requirements which are of primary domain importance: non-linear authoring and player freedom of action. These promote awareness of narrative potential while not ensuring it. Our *Creativity* metric item measures how well participants created this narrative potential; comparing it with measures of non-linearity provide some insights about this. We compared system and domain exploration with our logging of system usage and the task completion scores to gain information about how the design aids promote usage of VRBridge and when this usage is effective and focussed. The questionnaires completed the picture by providing evidence of end-user satisfaction with VRBridge. The components of the QUIS questionnaire provide evidence about satisfaction with various system aspects: overall enjoyment; understanding of domain-specific terminology; screen layout, sequencing and accessibility; and learning. In a similar manner, the components of the other questionnaires provide evidence about satisfaction with Timelines, Triggersets, each representation and each form of scaffolding. If end-users explore more effectively, perform better and enjoy the experience more in later sessions, then we have evidence about ease of learning with VRBridge. By combining these measures we triangulate our

results and gain a detailed picture of VRBridge and domain usage, and multiple forms of evidence about our success.

10.1.1. Evidence for the effectiveness of representations as a design aid

As discussed in Chapter 3, external representations enhance cognitive activities, especially memory and reasoning. The relationship between internal and external representations is known as *distributed cognition*. In approaching a task, the user employs the external representations as strategic aids, by visually perceiving and interpreting them and constructing internal strategies and conceptual models based on this perception. The representations direct attention, and provide external constraints and rules for correct action. We provide multiple, interactive, synchronised and simple domain-specific representations to assist design. It is important to recognise the significance of these attributes, as each adds power to the basic representations. Figure 10.2 displays the distributed cognition of the end-user supported by visual representations, showing the specific benefits of each attribute, and how they promote effective *lookahead*, appropriate biases and the formation and usage of learned knowledge during problem solving.

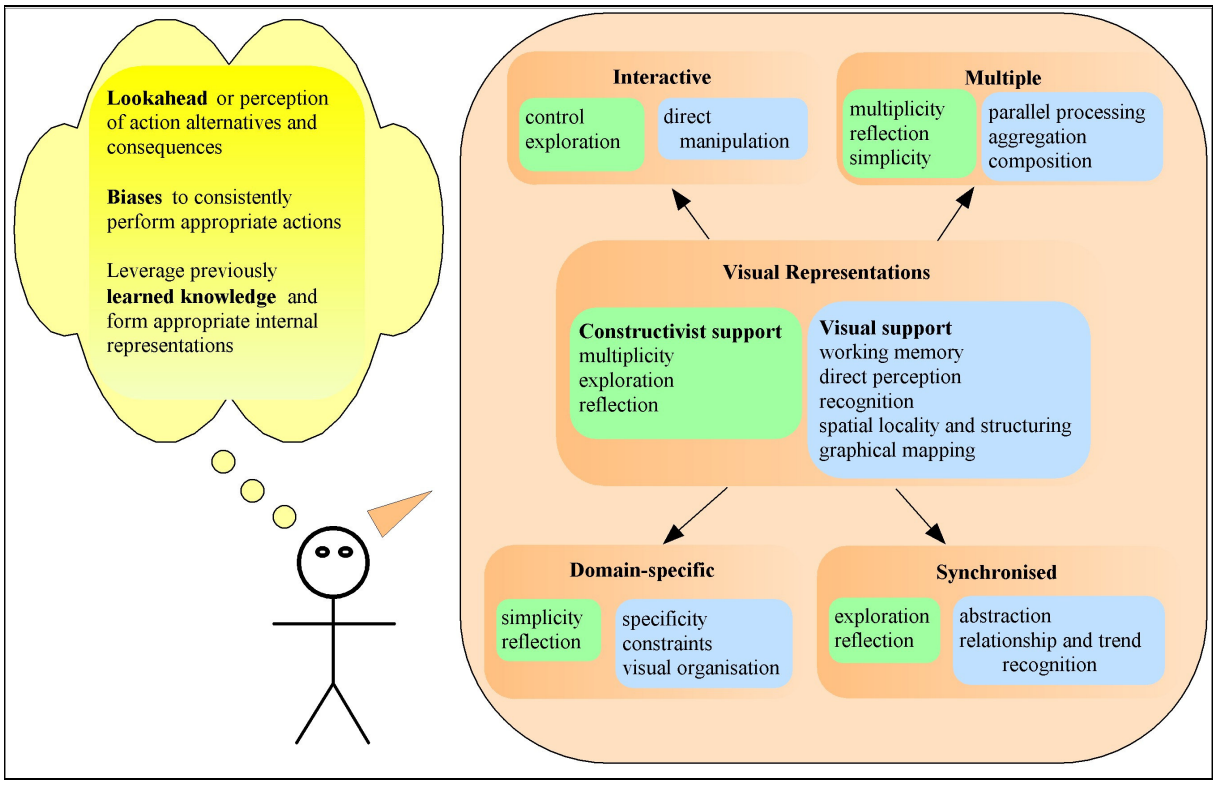


Figure 10.2: Distributed cognition of end-user with visual representations, showing the constructivist (green boxes) and visual (blue boxes) support offered by multiple, interactive, domain-specific and synchronised representations.

- Multiple representations provide the ability to compare and juxtapose aspects of a design. This attribute is based in the constructivist principle of *multiplicity* and fosters *reflection* by the end-user. Each representation can be *simpler* and easier to understand, as it does not have to convey multiple concepts.
- Interactivity helps the end-user to engage with representations directly and manipulate them. This fosters active and deeper *exploration*, and a sense of personal *control* in being able to manage the experience.
- Synchronising representations assists the end-user in recognising them as aspects of the same design. This aids *exploration* and *reflection*; as the end-user manipulates one representation, changes are propagated to the others, highlighting their connections and enabling awareness of the complex ways in which design aspects link together.
- We provide *simple domain-specific* representations, which are widely used and easy to understand, to convey domain concepts. Most end-users will have previous experience with floorplans, flow diagrams and timelines. Therefore, they can focus on understanding the domain skills and concepts conveyed by the representations.

In Section 3.1.3, we introduced the concepts of *informational* and *computational equivalence*. The information provided by our representations can be inferred from the basic authoring tool, meaning that they are informationally equivalent. However, our evaluations have shown that the representations are computationally more efficient, as information can be inferred more quickly and easily from them. This is discussed below.

Both of our evaluation studies provided evidence about the effectiveness of our representations as design aids, and the contributions of each. We chose the Floorplan, Timelines and Sequence Diagram as they corresponded to three important aspects of VE interaction design: usage of space and time, and non-linear sequencing of interactions. The first study (Chapter 7) clearly established that providing these representations improves end-user analysis of VE interactions: participants with representations understood the significance of interactions and their connections more accurately (by a third); and detected over twice as many errors.

The second evaluation study (Chapter 9) was more complex, providing us with multifaceted evidence about the effectiveness of our representations. It established that the provision of multiple representations significantly improves end-user task performance, exploration, satisfaction and learning. Participants with representations scored significantly better than those without on *Variable Sequencing* and *Player Freedom* (see Table 10.1), indicating that they learn and use important domain-level skills better. Participants with representations also scored significantly better on *Creativity* and *Features*, showing that end-users with representations have more interest in exploring narrative potential and playing with the system and domain, and do this better. For *Inevitable Sequencing*, those with representations scored less well than those without

(but not significantly) in all but the last task, where they almost doubled their previous scores. This, combined with their success at *Variable Sequencing* and *Creativity*, suggests that they focussed on important domain-level authoring skills initially, and broadened their skill to include less important authoring actions as they became more familiar with the system. These findings are supported by the logging results, which indicated that end-users with representations perform more exploratory actions.

When we examine the increased satisfaction of those with representations, we gain more detailed evidence about this aspect of performance. End-users with representations experience significantly more satisfaction with the overall system, the screens, and the domain-specific terminology used. They do not experience significantly more satisfaction with their learning. End-users with representations experience significantly more satisfaction with Triggersets initially, but this evens out over time. The results for terminology are particularly interesting. End-users with representations are more comfortable with domain-specific terminology, most likely because it is explained more clearly through the representations. They are even more satisfied than end-users who receive scaffolding, which more directly addresses terminology understanding. We conclude that the addition of visual images and the usage of terminology in the context of interacting with a diagram increase satisfaction significantly.

Usage of Representations

The two studies indicated how each representation was used and the ways in which they were useful. The first study established that end-users work comfortably with multiple windows and naturally compare multiple representations against each other. They also feel positive about the ability to interact with representations. In the first study, Timelines were included as an additional representation, whereas in the second study they were part of the authoring interface provided to all participants. We changed our definition of Timelines to be an authoring mechanism rather than a design aid for two main reasons:

- They are very useful for easily constructing interactions that must be sequenced together and so they are an integral part of authoring unlike the other representations.
- While they are useful for authoring, they can lead to programming more linear interactions. Since our design aids should foster skilful work in the domain, Timelines are not suitable in this respect.

We discuss Timelines as an authoring mechanism in Section 10.1.5. We added the Run Mode as a design aid to assist end-users in understanding how interactions link over time. In the second study, all representations (i.e., Floorplan, Sequence Diagram and Run Mode) were repeatedly used in both sessions, often for minutes at a time. This shows that participants found value in all of them, and suggests that they all contributed to the improved performance. Figure 10.3 depicts the differences in attention required, degree of interaction and degree of abstraction of the three representations.

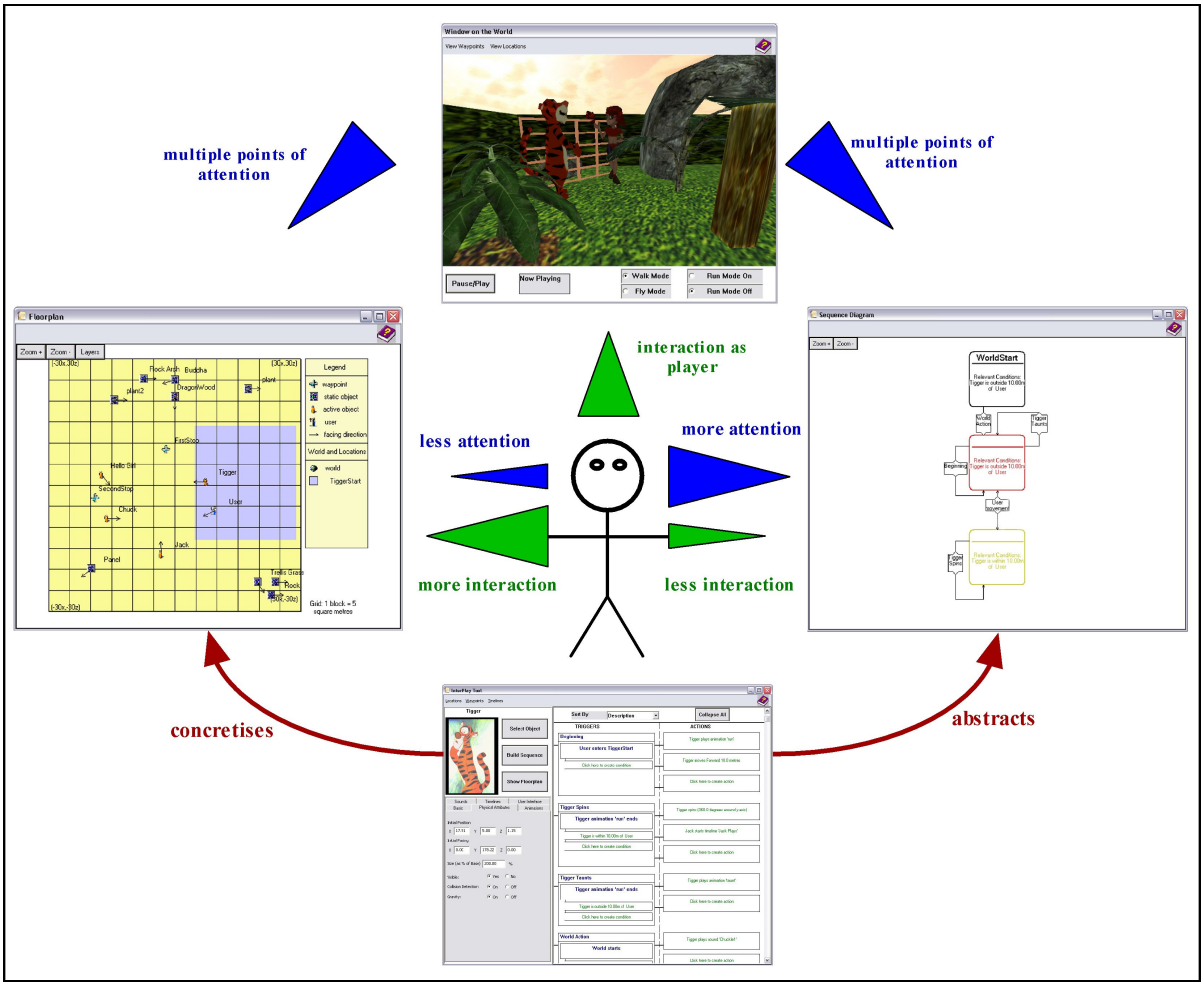


Figure 10.3: Graphical depiction of the differences in attention required, interaction provided and degree of abstraction of Floorplan, Sequence Diagram and Run Mode.

The Floorplan provides a valuable overview of the space, its objects and their relative positions; this adds meaning to the interactions as it gives them context. It makes it easier to work with the space through direct manipulation (e.g., for creating locations), and view the effects of manipulation. It provides a concrete visual representation of the relative positions of objects in the VE space, which is easier to perceive than the 3D world, even though it is informationally equivalent. It also requires less attention than the other representations, as perceptual principles such as enclosure and position can be pre-attentively processed to highlight relevant spatial information without focussed attention. Therefore it can be easily used while focussing on other design aspects.

The Sequence Diagram provides an overview of how the VE interacts with the player; end-users can visually confirm how the interactions connect and the context in which each is activated. In this way it visualises the

possibilities for narrative flow in the VE. It also supports end-users in thinking in the style of the domain, and being aware of its complexities. The Sequence Diagram is a useful debugging mechanism, as end-users can confirm their analysis of how interactions work together, and examine the diagram for indications that the results are not as expected. By manipulating the Sequence Diagram, end-users can identify where a set of interactions breaks down.

There are three main differences between the Sequence Diagram and Floorplan in terms of visual support:

- While the Floorplan concretises the space of the VE, the Sequence Diagram provides abstraction. It assists the end-user in thinking more abstractly about the Triggersets, viewing them in terms of connection and flow of control through the interactions and the VE, based on player input. This supports *reflection*, where the Floorplan supports *exploration*.
- The Sequence Diagram is less interactive than the Floorplan: end-users can create spatial markers and move objects using the Floorplan; they can only highlight interaction paths and states with the Sequence Diagram. Therefore, while end-users can interact with the Sequence Diagram, they can only use it to reflect on the implications of their designs, not perform authoring work.
- The Sequence Diagram requires more attention than the Floorplan. Although it also uses pre-attentive processing to highlight relevant connections, it problematises understanding, rather than simplifying it like the Floorplan. End-users must focus attention on the Sequence Diagram to examine how their interactions link together and connect with player interactions.

The Run Mode connects the Sequence Diagram, Floorplan and other features of VRBridge. Where the Sequence Diagram helps end-users to understand the consequences of interactions over time and context, the Run Mode allows them to view the design dynamically. It is connected to the operation of the 3D Window, and visualises a player's path through the interactions using all the VRBridge interfaces. The end-user can use this to understand the consequences of player interactions, and how the parts of the design connect together as a dynamic whole (through visualising the flow of action through the representations, Triggersets and Timelines). It is also useful for debugging: it connects the interactions more obviously to the player's actions in the 3D world, and makes it more obvious when a Triggerset is not executing. The Run Mode is similar to the Sequence Diagram in other ways: while it is dynamic, it is not interactive once started (except in that it can be paused and the designer controls the player in the 3D Window); and it requires focussed attention. In addition, since multiple windows are open and action is occurring in all of them, it can be confusing and requires selective attention to areas of interest. In this way, it problematises the design as does the Sequence Diagram.

The Floorplan was used most and earliest. As familiarity with the system and domain increase, the Floorplan is kept open longer, suggesting that it is used for constant reference and therefore is an integral design aid. It

is informationally equivalent to the 3D window and Object Panel, but is computationally more efficient and expressive. Conversely, the Sequence Diagram is kept open for less time, which indicates that it is more useful for initial domain exploration. Use of the Run Mode follows the same pattern as that of the Sequence Diagram, showing that they are likely used for similar purposes. Once the end-user understands how interactions trigger each other, and how the representations relate, the Sequence Diagram and Run Mode are less useful unless complexity increases. Interestingly, satisfaction with the Sequence Diagram increases over time, indicating that end-users become more aware of its usefulness even as they use it less. It is informationally equivalent to the Triggersets. When the Triggersets and their connections are simple and well understood, it provides no computational advantage. However, when the interactions become more complex or mistakes must be found, it is computationally more expressive, as it explicitly visualises connections which must be inferred from the Triggersets.

The reasons for the preference of the Floorplan over the Sequence Diagram can generally be understood by noting the differences shown in Figure 10.3. It requires little attention and is a visual representation that allows authoring actions (i.e., spatial design, such as creating waypoints and moving objects), unlike the Sequence Diagram. This shows the importance of interaction in end-user satisfaction. Interaction allows the end-user to control the process, work actively and feel satisfaction when goals are accomplished. In addition, we showed in Chapter 4 that novices prefer concreteness while experts prefer to be more abstract. The Floorplan concretises the space of the VE, while the Sequence Diagram abstracts the Triggersets. Initially, novices will prefer a representation that makes things more concrete.

The Run Mode and Sequence Diagram were not used very much during our final evaluation, and participants were not very satisfied with them. We could improve VRBridge by exploring how to make these more usable and friendly for novices.

10.1.2. Evidence for the effectiveness of scaffolding as a design aid

We provide multiple forms of non-invasive, user-adaptable scaffolding to support end-users in working with VRBridge and the domain. Figure 10.4 displays the distributed cognition of the end-user supported by scaffolding, showing the benefits of the attributes described above, and how they promote conceptual understanding of the domain, insight into expert strategies, procedural understanding of how to accomplish tasks, and metacognitive understanding of one's progress.

The goal of the scaffolding is primarily to support *exploration* and *reflection*. It supports *exploration* through explaining tool features and how they can be used. It supports *reflection* by explaining the link between authoring actions and domain requirements, and by showing how authoring actions can be enhanced. As with the representations, the scaffolding has attributes derived from our constructivist principles:

- *Multiple* forms of scaffolding allow us to address various authoring issues in a way that works best for each. Thus, we provide Tooltips for explaining terminology and how to work with VRBridge, Context Sensitive Hints for conveying domain knowledge and typical debugging issues, and Wizards for working through difficult authoring tasks. Tooltips and Context Sensitive Hints are *simple* descriptions; Wizards make complex tasks simpler by decomposing them and helping the end-user to focus on critical elements. End-users can use the scaffolding for which they have most need.
- Non-invasive and user-adaptable scaffolding keeps *control* in the hands of end-users. They choose when to open the scaffolding (except for Tooltips, which can be switched off) and when to stop using it. Their work is therefore not interrupted by the system.

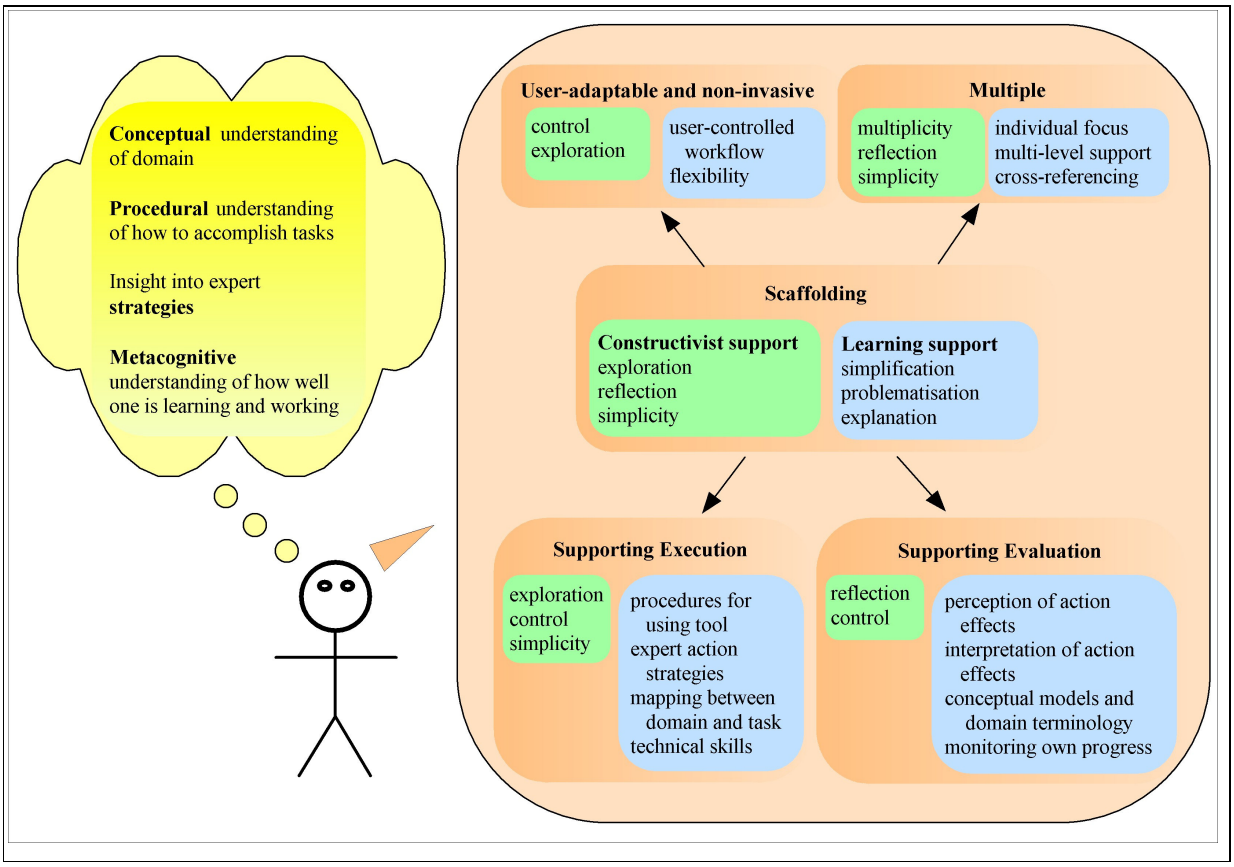


Figure 10.4: Distributed cognition of end-user with scaffolding showing the constructivist and learning support offered by multiple, user-adaptable and non-invasive forms of scaffolding.

Our first study provided information about areas in which scaffolding was desirable. These were primarily:

- 3D transforms, such as translation and rotation. Successful support would show in an improved *Creativity* score, as this measures more expert usage of space and interactions (see Table 10.1).

- Significance of the Sequence Diagram. Successful support would show in an improved *Variable Sequencing* score, as the Sequence Diagram fosters more expert creation and sequencing of Triggersets.
- Debugging interactions. Successful support would show in an improved *Task Completion* score.
- Using VRBridge and initial design steps. Successful support would show in improved scores for *Task Completion*, *XML Explore* and its components in the first Task and Session.

The second study provided evidence about the effectiveness of our scaffolding as a design aid, and the contributions of each form. It established that the provision of multiple forms of non-invasive, user-adaptable scaffolding significantly improves end-user task performance, exploration and learning. Participants with scaffolding scored significantly better than those without on *Variable Sequencing* and *Player Freedom*, indicating that end-users with scaffolding learn and use important domain-level skills better than those without. They also scored better on *Creativity* in all tasks, showing that end-users with scaffolding consistently show interest in playing with the system and domain. For *Inevitable Sequencing*, scaffolding did not significantly improve performance, but end-users with scaffolding tended on average to perform better than those without it. These results are supported by the logging, which indicated that end-users with scaffolding perform more exploratory actions. Therefore, the scaffolding supported ease-of-use and domain learning.

End-users with scaffolding experience significantly more satisfaction with domain-specific terminology. However, they do not experience significantly more satisfaction with other components; while their satisfaction tends to be higher, it is not significantly so. They do experience significantly increased satisfaction with the scaffolding over time; this increase indicates that end-users appreciate the assistance in recalling information when they return to the work after an absence. They value the scaffolding more than they did initially, when the tutorial and learning task had only recently been accomplished.

Usage of Scaffolding

The second study indicated how scaffolding was used and the ways in which participants found each form to be useful. All of the scaffolding (i.e., Tooltips, Context Sensitive Hints and Wizards) was used in both sessions. This shows that participants likely found value in all of the scaffolding forms, and suggests that they all contributed to improved performance. Figure 10.5 depicts the differences in attention required, degree of interaction and degree of sophistication of the three forms.

Tooltips are the simplest form of scaffolding. They provide the least interaction, as they appear automatically and are brief. They are *supportive scaffolding* and are used to explain how to work with VRBridge, perform authoring actions within the domain, and to define unfamiliar terminology. Therefore, they support procedural learning about the tool and steps required to complete tasks, and conceptual learning about

domain terminology and therefore insight into expert authoring strategies. They simplify the authoring task. Tooltips are both the most disruptive form of scaffolding and require the least attention. Although the user does not choose to open them, they can easily be ignored while working. End-users can pay attention to them briefly, to confirm the correctness of their actions or for a quick suggestion about how to begin.

Context Sensitive Hints are more sophisticated than Tooltips: they provide longer explanations of domain concepts; detailed steps for authoring actions; examples; and debugging hints. They are *supportive and reflective scaffolding*: they simplify the authoring task by structuring it, decomposing it into smaller steps and guiding users towards correct actions; however, they also problematise it by provoking reflection through highlighting domain strategies, explaining domain concepts and showing how these connect to the tasks and tool functionality. In this way, they support procedural, conceptual and strategic learning at a deeper level than the Tooltips. Hints provide slightly more interaction and require more attention than Tooltips; the user must explicitly request support by opening a Hint.

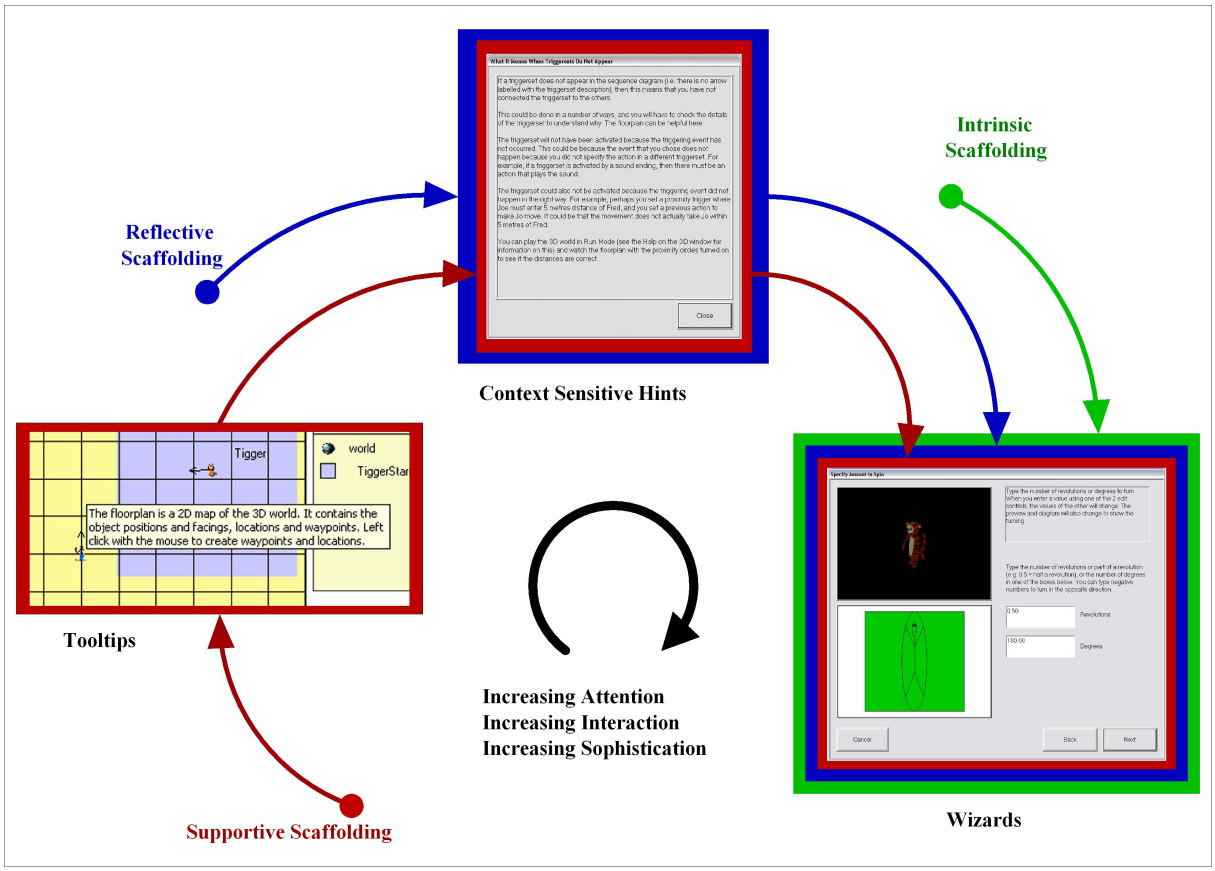


Figure 10.5: Graphical depiction of the differences in attention required, interaction provided and degree of sophistication of Tooltips, Context Sensitive Hints and Wizards.

Wizards are the most sophisticated form of scaffolding that we provide. They are *supportive, reflective and intrinsic scaffolding*: they simplify authoring by providing step-by-step guidance about how to perform tasks in 3D; they problematise it by provoking reflection through explanation of domain concepts involved in task steps; and they replace the task with a scaffolded version, rather than interrupting it. We provided Wizards for movement and rotation in 3D. The Movement Wizard was more complex and more obviously helped users to work with the space by moving objects across it effectively (rather than just performing rotations); it was used twice as often as the Rotation Wizard. Wizards are the most interactive form of scaffolding, as the user performs tasks by working with them. Although they slow authoring down and require the end-user to change context (from the Triggerset to the Wizard interface), the Wizards add concrete value to tasks as opposed to advice alone. They do this by providing a safe environment to practice authoring (as mistakes can be easily deleted and are not entwined with the Triggersets), and by allowing end-users to add sounds and animations to the action, and in the case of movement, rotation to face the direction of travel. Therefore, the Wizards support conceptual, procedural, strategic and metacognitive learning.

Fading of Scaffolding

Our results for the fading of scaffolding (i.e., lessening or stopping its usage) were not successful and must be viewed with the proviso that participants only experienced two sessions with VRBridge a week apart, comprising a maximum of three hours (and received external assistance during the first task). This time period is not sufficient to evaluate fading completely, and that fact is shown in our results. We do find some interesting differences in how the scaffolding forms are faded:

- Tooltips are faded by turning them off. However, only two of our participants did this. As they are the most disruptive form of scaffolding, this shows that end-users value their brief reminders. However, because they disappear after a few seconds, they can also be faded by being ignored, and this result was not logged; therefore we cannot conclude much about Tooltip fading, as our measures did not capture its usage.
- Context Sensitive Hints were faded the most. Participants used them much less during the second session. It is likely that end-users would remember the information in the Hints after brief reminders from the Tooltips, and therefore not have to reread them. Supporting this theory is the fact that the only Hints not faded by the Scaffolding group were those on the Main Menu, which describe how to start, and remind the user about VRBridge features and basic domain requirements. With this information, end-users can begin re-exploring the system and remind themselves about how to proceed. However, the lack of Hint usage could be because they must be opened and are mostly textual, requiring focussed attention.
- The Wizards are the scaffolding form that increased most in subjective value over time. Satisfaction with the Wizards increased significantly in the second session. In addition, participant usage of Wizards generally increased as opposed to fading. In the time that participants worked with VRBridge, the Wizards made authoring simpler and their constraints would not yet be considered too restrictive. End-

users were not likely to have gained sufficient expertise that they could complete tasks faster and more precisely without the automation that Wizards provide.

The lack of fading of the Wizards is connected to the fact that they are intrinsic scaffolding and the most sophisticated and interactive form. Jackson et al. (1998) define supportive, reflective and intrinsic scaffolding (see Section 3.2.1) and state that intrinsic scaffolding is the most difficult form to fade and they are unsure about how this would be done. While our research supports their finding, we can provide more precise reasoning. The more sophisticated forms of scaffolding may never fade or will fade over a longer period of time than others. This is supported by the fact that the Combination group did fade the Rotation Wizard, which is less sophisticated and obviously useful than the Movement Wizard. Scaffolding that is task-based, and therefore integrated with the work, is useful for longer and provides more satisfaction than scaffolding which only offers advice.

The value of scaffolding is that it fades and the user is able to complete the task independently without relying on support. End-users did learn with the scaffolding, and improved their *Variable* and *Inevitable Sequencing* scores, showing that their expertise increased. However, they still relied on the support of the Wizards. Therefore, we did not accomplish our aim of creating scaffolding which users would fade.

However, since Triggersets are not connected to the Wizards (as Wizards output Timelines), this shows that authoring without Wizards improved, and so users were not negatively impacted by their continued usage. Therefore, more sophisticated, task-based scaffolding should be carefully constructed so as not to hinder development if users choose not to fade it. Our connection of Wizards to Timelines rather than Triggersets acted to prevent over-usage. Our findings also show that end-users customised the Timeline output of Wizards more in the second session, as their *Inevitable Sequencing* scores improved dramatically, showing more expert usage of Timelines. While this is not fading, it does show that end-users worked more independently of the Wizards; they likely used the Wizards to easily create the structures of basic Timelines, and then added to these to make them more effective. Care must be taken with integrated, more sophisticated scaffolding so that ongoing usage does not hinder the development of expertise, as this scaffolding may never be faded.

10.1.3. Comparing and combining representations and scaffolding

An important part of our research was to compare the relative effectiveness of additional representations and scaffolding for supporting effective authoring, and to evaluate the combination of the two. As we established above, both representations and scaffolding add much value to the basic authoring tool. Figure 10.6 graphically depicts the main benefits offered by each and their combination.

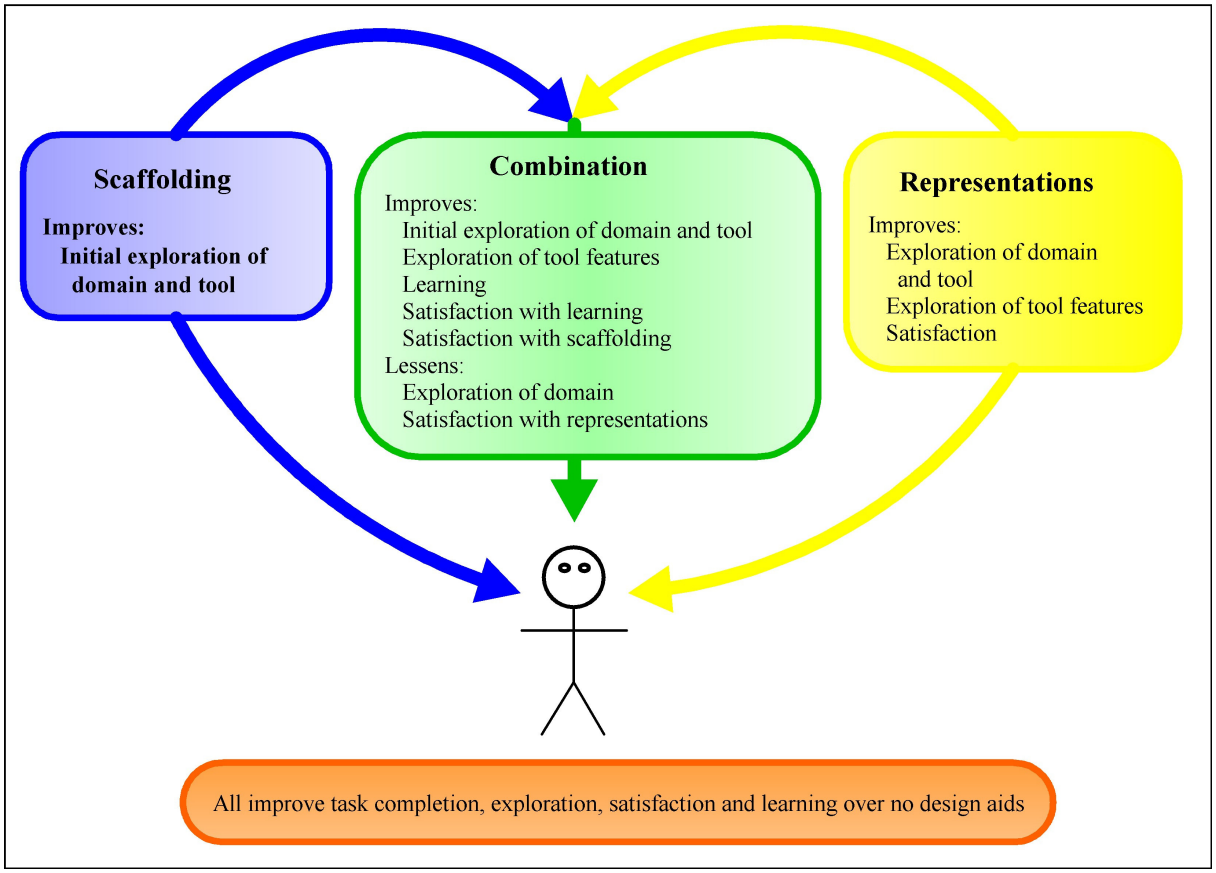


Figure 10.6: Benefits offered to the user by Scaffolding, Representations and their Combination.

When we compare relative effectiveness, representations show broader value than scaffolding, although there are only two significant differences between the two:

- End-users with representations feel significantly more satisfaction overall; and
- They feel significantly more satisfaction with domain specific terminology.

For the other satisfaction measures (i.e., satisfaction with screen, learning, Triggersets and Timelines), those with representations show a tendency to be more satisfied than those with scaffolding in all but Timelines, but the differences are not significant. The result for terminology is interesting, as end-users with scaffolding have more explicit help in this regard. However, as noted above, representations provide a visual connection between terminology and what it denotes; this would increase end-users' confidence in their understanding and be more enjoyable than reading definitions. Unlike our satisfaction results, while end-users with representations tended to perform better at *task completion* and *exploration*, there are no significant differences. The evidence for exploration leads us to conclude that representations do aid more in this regard than scaffolding, even if the differences are not significant for our study. End-users with representations improve the most in successful exploration, beginning as bottom scorers and ending as top scorers. They are

the top scorers in *Variable Sequencing* and equal to those with scaffolding at *Player Freedom*, our core design skills. They also generally perform more exploratory actions with VRBridge than end-users with scaffolding.

Where the scaffolding provides more benefit than the representations is in initial explorations. End-users with scaffolding are the top scorers in Task 1 for *exploration*. They also show the greatest *Creativity*, and have higher scores for *Variable* and *Inevitable Sequencing*. This establishes that scaffolding improves initial tool usage more than representations, but the advantage disappears over time. Those with scaffolding are not the top scorers in skilful *task completion*; however, since VR design is a highly visual task, the benefits provided by the visual representations offset the initial scaffolding advantage. As scaffolding is meant to fade, the fact that its benefits fade over time, unlike those of the representations, is appropriate.

It is interesting that such different design aids have similar effects. We can understand this seeming contradiction more clearly when we consider that both scaffolding and representations provide support; whatever the form of this support, it adds value to the experience. Representations add more long-term value. When we examine the effectiveness of the combination of representations and scaffolding, we find an unexpected result. The benefits of the design aids do not increase substantially when combined; in fact, they decrease in many cases. This is in contradiction to our hypotheses based on the constructivist world view that more perspectives and assistance automatically improves learning and understanding. End-users who receive both representations and scaffolding *complete tasks* significantly more skilfully than those who receive nothing; but equally to those who receive either representations or scaffolding. For *effective exploration*, end-users with both design aids perform slightly better than those with representations in the first task. This supports the result for scaffolding usefulness in early explorations. However, they perform worse than those with scaffolding or representations in subsequence tasks.

Those with both design aids perform significantly better on *Variable Sequencing* and *Player Freedom* than those with no design aids, and better on *Variable Sequencing* than those with scaffolding. However, for *Inevitable Sequencing*, they score significantly lower than those with scaffolding. The results on *Variable Sequencing* show the benefits of the Sequence Diagram for sequencing Triggersets, and that those with both design aids learned important domain skills. However, the low score on *Inevitable Sequencing* is less easy to untangle. One possible reason is that end-users with both design aids played briefly with Timelines, but chose not to develop them and therefore scored lower. This interpretation is supported by their low Timeline satisfaction. End-users with representations did not have access to the Wizards, so would not have played with Timelines as casually. Another reason is that end-users with both design aids have more information to assimilate. This would lead them to explore broadly and less effectively, and would lessen the attention paid to less important authoring aspects. As Triggersets were highlighted as the core authoring interface in the

scaffolding, and supported by the Sequence Diagram, they would have more reason to work with them. The fact that they scored lower on *Creativity* than those with a single design aid also supports this theory, as they would have less attention available for anything but the basic task.

The area where the combination of scaffolding and representations improves results is *learning*. End-users with both design aids increase their exploratory actions more than any others and were most satisfied with Triggersets in the second session. Although they do not explore more effectively, they improve their task scores the most. They express the most satisfaction with their learning, and are significantly more satisfied with their learning than those without design aids. For overall and screen satisfaction, they are significantly more satisfied than those without design aids and more satisfied than those with scaffolding. However, for satisfaction with terminology, they are significantly less satisfied than those with representations alone, and equivalent to those with scaffolding. This is an interesting finding, as it seems likely that adding diagrams to terminology explanations improves satisfaction. However, for those with both design aids, the support for understanding terminology is not integrated. Since the scaffolding defines terminology more explicitly than the representations (through textual descriptions) the Combination group probably associated their terminology satisfaction with the scaffolding.

End-users with both design aids are slightly less satisfied with the representations than those who only received representations. They are significantly less satisfied with the Run Mode. This evidence supports our theory that the combination of design aids means that their attention is overwhelmed. The Sequence diagram and Run Mode require significant attention to achieve design benefits, compared to the Floorplan, which is more obviously and immediately useful. When we examine satisfaction with scaffolding, we see that those with the combination feel similar satisfaction with the scaffolding in both sessions with VRBridge, while those with only scaffolding feel more satisfied with it later. In addition, initially those with only scaffolding feel less satisfied with it than those with both design aids. End-users with both design aids appreciate the benefits of the scaffolding immediately, as shown by their satisfaction with learning, probably because they see its benefits in using the representations.

The effectiveness of the combination of representations and scaffolding is only compellingly seen in increased satisfaction with learning. This is an important contribution, as end-users who are satisfied with their learning will be more likely to continue working with the tool and domain. However, they do not show increased satisfaction with other parts of the system and in some cases, show more frustration. Care must be taken in adding design aids: as well as providing benefits, they act to decrease the simplicity of the authoring interface and therefore increase the complexity of the task. Although performance is improved from the basic case where no design aids are provided, in many cases the improvement is less than when only a single type of design aid is provided.

10.1.4. VRBridge as successful embodiment of constructivist principles

We have discussed how the design aids embody our constructivist design principles. We have also established that they improve tool usage, domain understanding, task performance and learning. Therefore, according to the metrics described at the beginning of this chapter, we have shown that constructivism provides a practical basis for structuring and supporting VE interaction authoring with design aids. We have also shown that there are concerns with adding too much information and overwhelming the novice user.

We also used constructivism in designing our basic tool, VRBridge and its event-based authoring interface, Triggersets. Here we consider evidence for the overall effectiveness of VRBridge without design aids. The basic system is characterised by the following features: a Triggerset interface with dialogs and natural language descriptions; Timelines; an Object Panel with textual interfaces for changing object details; Previews of sounds, animations and models; a 3D window, with options to fly or walk, pause, and view waypoints and locations; and Textual interfaces for creating locations and waypoints.

We have shown that end-users who are novices in VR authoring can use VRBridge to create relatively complex combinations of typical interactions. Even those who received no design aids were able to complete tasks and improve their task performance, tool exploration and satisfaction in a subsequent session. This means that VRBridge enables users to do creative work in a field where programming expertise is typically needed, and begin learning necessary domain concepts (like defining actions precisely, using if/then statements, and player freedom) and terminology. Satisfaction is highly desirable as, if early experiences are fulfilling, users are more likely to continue exploring the tool and domain. Users can create dynamic interactions between objects in a 3D world and explore these actively themselves; as they learn about 3D interaction design, they can immediately design and create an artefact which embodies this knowledge. This aids reflection, promotes control of one's learning process and therefore increases satisfaction. In addition, as users learn the terminology and constraints of the domain, they have the resources for asking appropriate questions to learn more. We provide the ability to prototype early, as all users are able to create interactions in a VE within half-an-hour of working with VRBridge. This provides immediate feedback about the design and the ability to communicate it more effectively with a prototype.

Therefore, we have shown that constructivism serves the VE interaction design process, as the tool and design aids which we created based on investigation and application of core constructivist principles were highly effective in supporting successful authoring.

VRBridge Authoring Interfaces: Triggersets and Timelines

Here, we discuss the two authoring mechanisms in more detail; we compare and contrast usage of Triggersets and Timelines. Both were designed to foster the composition of simple atomic actions into complex interactions; and both use visual mechanisms to make authoring easier. The Triggersets were also designed to promote non-linear authoring, because of their disjointed nature: every Triggerset (and component action, trigger and condition) is a separate unit, which can be connected in various ways to other Triggersets and player actions. The Timelines were designed to make complex sequencing of interactions in time easier. They promote skilful synchronisation by providing a visual arrangement of actions in time, which can easily be adjusted and fine-tuned. However, this design choice negatively affects non-linear authoring and player interaction, as all the actions on a Timeline will occur in the order in which they were programmed, with no input from the player. Therefore, they do not help users to think non-linearly.

Usage of Timelines provides evidence about usage of the primary alternative authoring mechanism, and highlights the different authoring styles promoted by the design aids. End-users who did not receive representations were most positive about the Timelines and used them most skilfully. Conversely, those who received no other design aids were least positive about the Triggersets and skilful using them. They scored the worst on *Variable Sequencing* and on *Player Freedom*, which indicate domain-skilful authoring (see Table 10.1). While end-users with scaffolding preferred the Timelines and used them most successfully, they were able to use the Triggersets almost as successfully as those with representations; they did not score significantly worse on *Variable Sequencing* and scored equivalently on *Player Freedom*. They also scored equally well (and initially higher) on overall *exploration* and *task completion*.

This evidence provides support for two conclusions about the impact of our design aids on Triggersets and Timelines. The provision of representations adds enjoyment value, context and constraints to the Triggersets. The Floorplan and Sequence Diagram provide context for interactions, and therefore constraints. The end-user without these representations must open the 3D world to view the context of Triggerset interactions. Fly Mode can provide an overview of the space; however, this view is not symbolic and manipulable like the Floorplan. In the same way, the end-user can execute the interactions in the 3D world, but will not be able to view them in a single overview with the Sequence Diagram. In addition, the Sequence Diagram provides support for using the Triggersets effectively. The Timeline provides some of these constraints: end-users can view actions in the context of the other Timeline actions and the time scale. Therefore, the end-user without representations will spend more time and energy working with the Timelines as they are easy to understand, and provide immediate feedback about interaction sequencing. The end-user with other representations will not feel this need, as it has been fulfilled elsewhere.

Our second conclusion is that the provision of scaffolding almost completely alleviates the negative effect of Timelines on non-linear authoring and player interaction. Without the representations, Timelines are more likely to be used than Triggersets as they provide more contextual and constraint benefits. Without the scaffolding, this leads to less expert work. However, when scaffolding that highlights expert authoring practices is added, these negative effects are alleviated. Conversely, users who receive representations will use Timelines less and less well, even if they have scaffolding. This is likely because they view Triggersets as the main authoring mechanism, and the representations support expert usage in these.

As stated earlier, there were several kinds of interaction that could not be easily captured with Triggersets: e.g., probable actions; actions associating objects and actions attached to multiple objects. In addition, as we have seen above, the addition of Timelines to deal with time-based actions, made our interaction system less effective at promoting expert authoring. These problems must be fixed for our Triggerset system to be completely usable for creative work. While the design aids address some of the problems with the system, e.g. promoting non-linearity despite usage of Timelines, ideally the Triggerset system must be able to address these issues itself, so that it has enough expressive power to be useful as novices become more expert.

10.1.5. Synthesis

We have shown that the form of authoring mechanisms and design aids determines the information perceived by the end-user. A feature's form provides constraints on correct actions: the Triggerset form suggests non-linearity while the Timeline form suggests sequencing. The Floorplan provides constraints on how space is used. We can support more effective usage of a system and expert work in a domain by leveraging these constraints. Table 10.2 indicates the implications of our research for VRBridge features and design aids.

System Part	**Features and Implications for Support**
Basic Authoring Tool	
Triggersets	Support reflection on usage for effective domain actions with Sequence Diagram
Timelines	Support understanding of limitations Promote non-linear usage with Sequence Diagram
Object Panel	Support visualisation of space and objects with Floorplan and previews
3D Window	Support overview of space and objects with Floorplan Support connection with authoring and player perspective with Run Mode
Waypoints/ Locations	Support direct manipulation with Floorplan
Representations	
General	Add more interactivity

System Part	Features and Implications for Support
	Where more attention is required, support effective focus
Floorplan	Support effective usage with scaffolding
Sequence Diagram	Make more interactive Support effective usage with interactive scaffolding Make simpler to understand and learn from
Run Mode	Explain usage with scaffolding so less overwhelming, e.g., Wizard showing typical usage
Scaffolding	
General	Integrate visual elements for more satisfaction Add more interactivity Support safe initial learning Provide supportive, reflective and intrinsic scaffolding
Tooltips	Provide scaffolding which requires only brief attention, for quick reminders
Context Sensitive Hints	Integrate visual elements for more satisfaction
Wizards	Wizards are useful for difficult design areas, e.g. Run Mode, initial authoring Ensure output can be customised so expertise develops even if usage continues Ensure support of domain expertise so ongoing usage has no negative effect
Combination	
General	Scaffold initial steps and encourage play so less overwhelming initially Highlight importance of creativity within the domain Manage appearance of screen to control attention required

Table 10.2: List of VRBridge features and design aids, showing the implications of our research for each.

We confirm the effectiveness of VRBridge by examining our results against the guidelines and findings by experts described in Section 4.1.1.

- Experts use domain knowledge effectively in solving problems. End-users who received our design aids showed increased skill at creating interactions that embody domain values. They were able to integrate the knowledge provided by VRBridge into their reasoning and apply it effectively.
- Experts create and use multiple representations effectively and repeatedly to solve problems. VRBridge users with representations were able to manipulate them, and use them to create effective interactions and evaluate their designs. Because our representations are simple, it is likely that VRBridge users will be able to internalise them and create their own representations when addressing similar problems.
- Experts find patterns and perceive important features; they are less likely to recall unimportant information. Our first study showed that those with representations were far less likely to note

unimportant interaction details; our second study showed that those with representations and scaffolding explored VRBridge features more effectively and applied them more appropriately to the task.

Limitations of our research are shown in the difficulty that novices had working with the more complex representations and working with the combination of representations and scaffolding. In promoting multiple perspectives, the constructivism method can lead to the attention of the novice being overwhelmed by the apparent complexity of the task. Care must be taken to manage the appearance of multiple design aids. More investigation must take place so that the computer can more effectively take on the role of tutor to complete novices.

10.2. Implications for Supporting End-User Authoring

In this section, we discuss the generalisability of our findings and their implications for end-user authoring. We have shown that VRBridge supports novice authoring of VR interactions. Our metrics for effectiveness are themselves generalisable: the provision of a tool that is easy and fun to use and learn, and enough power to satisfy the end-user; and the support of domain-level learning of requirements for successful authoring, and therefore development of expertise. In terms of Blackwell's attention investment model (2001c, 2002), we reduce the costs of investing attention with successful provision of the requirements for the first metric, and increase the benefits to the end-user through successful provision of domain-level knowledge. Figure 10.7 shows how VRBridge and the design aids reduce costs and increase benefits of learning.

We have practically implemented Shneiderman's (2000) guidelines for enabling creativity with computers (see Chapter 1). We enable visualisation of work process and its artefacts through our representations; we provide immediate feedback to end-users so that they can explore, reflect on and evaluate their creations; through this, they learn to associate important domain concepts with practical work. We support this further by enabling access to expert knowledge through the scaffolding and representations. In Chapter 3, we described Jonassen and Carr's (2000) conception of the computer as a mindtool and their description of various kinds of support that computer tools provide. We take their idea further and suggest that to utilise the power of computers completely, we must encompass multiple forms of support in a single tool.

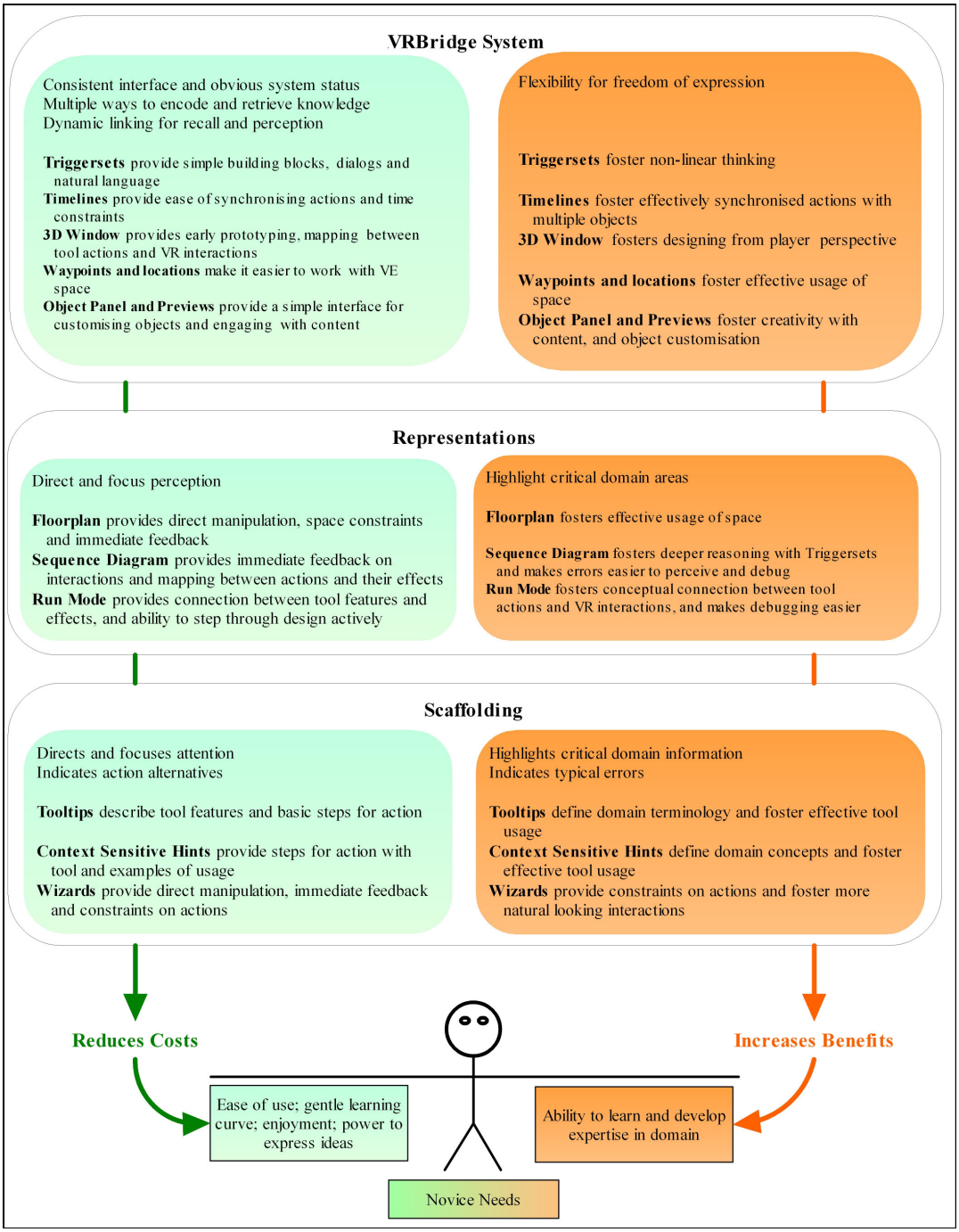


Figure 10.7: The ways in which the VRBridge system, Representations and Scaffolding reduce costs and increase benefits of learning new software and domain to the novice user.

Using Jonassen and Carr's taxonomy, we discuss the multiple forms of support that VRBridge provides, as evidence for our proposal:

- *Semantic organisation*: VRBridge helps end-users to organise and analyse their knowledge by practically applying it in the creation and evaluation of an artefact. It provides representations which indicate the mapping between interactions and their effects, and the structural relationships of the VE.
- *Dynamic modelling*: VRBridge helps end-users to describe and interact with the dynamic relationships within a VE. It provides a constrained, safe environment in which to explore and model relationships.
- *Visualisation*: VRBridge helps end-users to reason visually, both within the domain of VE interactions, and more broadly about programming and design. The entire interface is visual, as even the textual Triggersets are organised with visual connections between triggers, actions and conditions.
- *Knowledge construction*: VRBridge helps end-users to construct artefacts that encompass the knowledge that they learn; and therefore to learn by working actively within the domain.
- *Social sharing*: Although VRBridge is not a communication system, it enables communication and collaboration by making end-users aware of domain terminology and constraints. With these tools, they can communicate more effectively with others.

In combining elements of all these types of systems, VRBridge provides increased explanatory and exploratory power for end-users as they actively construct their knowledge. In a sense, we are supporting deliberate practice in the development of expertise by promoting active work within the field of interest, where performance can easily be monitored and improved through immediate feedback. We provide the following general guidelines for supporting end-user authoring, based on our research:

1. Provide simple domain specific but commonly used visual representations to guide the end-user and increase expertise. These can be internalised and recreated by the end-user.
2. Allow as much interaction with the representations and other parts of the interface as possible, providing advice where necessary. Being able to work practically increases knowledge uptake and effective usage.
3. Create an attractive environment which will inspire design ideas. With initial explorations in authoring, it is important to spark interest in the benefits of the process. Some provision of interesting content with which to work will achieve this, together with simple suggestions for how to begin. It will also increase enjoyment, as end-users will be able to create their own designs using professional content.
4. Enable the end-user to create a product as quickly as possible. Even if this is very simple, end-users will immediately have feedback about their work and proof of success.
5. Provide multiple forms of support carefully, so as not to make the interface too complex. With multiple supports, end-users can tailor their working environment and process to suit their particular learning style. Visual and textual supports should be integrated.
6. Where focussed attention is required, scaffold the process through interactive examples of how to work and highlighting of areas on which to focus, so that the end-user does not feel overwhelmed.

7. Provide a multi-functional tool: this will help users to gain appropriate new knowledge, organise it into effective structures, work practically with it to make it easier to retain, view it in different ways to understand all of its implications and aspects, and share it with others through artefacts.

10.3. Implications for interactive design systems

Another way in which we can generalise our results is by applying them to interactive design systems. In Section 4.1.3, we identified problems of interaction designers working with current tools. The most important of these were: lack of understanding technological constraints; difficulty prototyping because of complexity; constrained work process which impacted creativity; lack of ability to specify details appropriately for engineers; lack of linked representations which foster consideration of multiple facets of a design; and predefined interaction options to specify behaviour which were not sufficiently flexible.

Many of these problems are based in the fact that interaction designers cannot control the experience of the end-user; the design work therefore entails trying to foresee as accurately and completely as possible how to support any action of the user. Not understanding the technological constraints of the systems increases the difficulty. VRBridge addresses all of these problems for VR interaction design by creating a space where designers can think about their designs and experience them from the perspective of the player.

Below, we provide guidelines for creating interactive design systems, based on our research and experiences with VRBridge.

1. Provide a simple authoring interface that highlights important technological considerations, so that designers are aware of technology constraints.
2. With a simple authoring interface that delivers a basic but functional prototype, designers can demonstrate (rather than describe) interactions to engineers in enough detail to retain creative control.
3. Provide flexibility for how the design work will be accomplished, to allow different work processes to be followed. There should be minimal required order of implementation. Where this is necessary, simple place holders and work-arounds should be available.
4. Provide multiple, linked representations, which are appropriate to the design task, and allow viewing of facets of the design at multiple levels of granularity. Two types of representations are particularly useful: an overview of the design space so that interactions can be designed in context; and an overview of the interactions that have been designed, showing how they connect to each other and change the state of the system, and indicating where the end-user can influence them.
5. Representations should be connected dynamically so that the progress of the design and its interactions can be examined as they occur over time. This will also help with debugging interactions, as the place where interactions diverge from what was expected can be easily found.

6. Providing predefined options for interaction and content is valuable. However, it should be provided as flexibly as possible. For interaction options, providing a few simple building blocks which can be combined flexibly and powerfully is more useful than providing more complex predefined options. Although it might take more effort to create complex interactions, the power and activity remains in the hands of the designer; this gives the designer more creative control. Provision should be made for saving complex interactions that designers have created, so that these can be reused.
7. Where knowledge support is provided, it should be tightly integrated with the working environment. It is better to provide a designer with a Wizard to work through than a detailed document. The Wizard can highlight technological constraints while helping the designer to create a simple interaction; designers will learn more effectively about how to consider technology in the context of their designs.

11. Conclusion

This dissertation presents a constructivist design method for end-user authoring systems. Our domain of interest is VR interactions, and therefore we also present guidelines about the design of interactive systems. We applied constructivist theory at two levels: to structure our design method in creating an effective tool for supporting end-users; and to structure the design process of our end-users in the domain of VR authoring. We explored the use of design aids in the form of representations of the design process and scaffolding, and presented guidelines based on this research for their use. We conducted three studies to understand novice needs and evaluate the success of our authoring system and design aids.

11.1. Achievements

In Section 1.2, we presented five aims of this project and associated research questions. The achievements under these headings were:

1. *To develop and follow a theory-based design method which structures the creation of an interaction authoring tool*. We defined a novel method that structured an iterative design for system building, grounded in theory. We followed the method during the course of this thesis, conducting extensive research into the requirements of the domain of VR authoring, the practices of experts and how we could foster these, and the practices of novices and how we could address their difficulties.
2. *To identify and explore key constructivist principles that can practically be applied to an end-user interaction authoring environment*. This aim corresponds to our first research question: *What features would a VE authoring environment have if it were created with constructivist principles?* Through careful examination of the constructivist theory, we identified five core design values: multiplicity, active exploration, simplicity, user control and reflection. Based on these principles, we identified two features to develop as design aids: multiple, synchronised, interactive domain specific representations; and multiple forms of user-adaptable non-invasive scaffolding. We then conducted an extensive review of the literature on representations and scaffolding to distil guidelines on effective usage.
3. *To create a VR interaction authoring tool for end-users based on the features that we identified in (2).* We created VRBridge, a VR interaction authoring tool for end-users. It embodied the principles of providing multiple perspectives on the design and methods for achieving it, fostering extensive and active exploration of tool and domain, keeping the basic parts of the system and building blocks for interactions as simple as possible, providing the end-user with as much control and flexibility as possible while guiding the learning process, and fostering reflection on the work. While the design aids were the primary focus of our research, the authoring interface, which consisted of a visual organisation of trigger-condition-action triads or Triggersets, was also important.
4. *To evaluate VRBridge and compare the features identified in (2) in terms of how effectively they improve*

tool usage, domain task performance and conceptual understanding. This aim corresponds to our second research question: *What is the relative effectiveness of dynamic scaffolding and multiple interacting representations for making a design tool easier to learn to use and fostering conceptual understanding of the VE interaction domain?* We conducted two experiments to evaluate VRBridge in separate design iterations. The first evaluated the usefulness of the tool for analysing existing interactions, and focussed on the Triggersets and representations. The second was a large scale experiment (124 participants), which evaluated the basic tool, the representations, the scaffolding and their combination. As part of the second experiment, we developed a comprehensive metric for evaluating task success, tool and domain exploration and domain understanding.

5. *To evaluate how well constructivism serves the design process for a VE authoring tool, and the extent to which we can generalise these findings to other interaction authoring and end-user design tools*. This aim corresponds directly to our third research question. We investigated it through our evaluations of VRBridge and the design aids, as they had been created according to constructivist values. Our evaluations showed that constructivism serves the design process well, as end-users were able to create interactions easily with little previous experience; and the design aids supported increased authoring expertise. However, the attention of novices must be carefully managed so that they are not overwhelmed by the multiple perspectives that constructivism promotes. Following this, we reflected on our research and defined general guidelines for successfully creating interaction authoring and end-user design tools.

11.2. Summary of Design and Evaluations

We conducted three studies and created an authoring tool with design aids for end-users. Our studies provided a variety of data to evaluate end-user needs and our success in meeting them. They also provided information about how representations and scaffolding can most effectively support end-user authoring (see Chapter 10 for reflections on our results).

11.2.1. Novice problems and requirements for support

After examining the research on VR authoring requirements, the practices of experts and how to support novices, we conducted an exploratory study using classroom observation of novices designing a 3D game. We focussed on usage of programming constructs, the interaction design process and interactions with visual design aids. We wanted to find out how well novices understood programming constructs, their strengths and weaknesses in interaction design, and which design representations worked best for them. We found the following:

- Novices have problems understanding and implementing programming constructs in general, but use *if…then* statements naturally to express interactions.

- The greatest problems with interaction design are a tendency to produce linear designs with few alternative interaction possibilities for the player, and forgetting to document all details explicitly.
- The representations that are most effective and easiest understood are the floorplan and flowchart. Although novices produce inaccurate and linear flowcharts, they understand flowcharts well.

11.2.2. The VRBridge authoring system

Based on our research and guided by our constructivist ideals, we created VRBridge, an authoring system for interactions in 3D worlds. VRBridge has a modular object architecture which is extensible and allows a flexible workflow. Its authoring interfaces consist of the InterPlay screen for creating interactions with dialogs; the Object Panel for changing and viewing object details, such as starting position and attached sounds and animations; the Waypoint and Location Editor for creating spatial markers; and the 3D Window for viewing the 3D world as the player or from a bird's-eye view.

Interactions are created primarily using an event-action programming paradigm, with natural language and direct manipulation. We developed trigger-condition-action triads, or *Triggersets*, for authoring interactions. Triggersets consist of simple building blocks (triggers, conditions and actions) covering typical VE actions and events, which can be flexibly composed to form complex interactions. Timelines are an alternative authoring interface, and assist in skilful and easy synchronisation of interactions in time; however, they can lead to more linear authoring.

We added two design aids to the basic system: representations and scaffolding. The representations consist of a Floorplan to support understanding and manipulation of 3D space, a Sequence Diagram to foster non-linear interaction authoring, and a Run Mode, which dynamically shows the progression of interaction through the representations and authoring interfaces as the player interacts with the 3D world. The scaffolding consists of Tooltips for low-level, brief and supportive advice about the system and domain; Context Sensitive Hints for more detailed supportive and reflective advice about the system and the domain; and Wizards for intrinsic, task-based support for accomplishing difficult 3D interaction tasks.

VRBridge features are dynamically linked through context menus, highlighting and slaved views. The system design was structured by our constructivist principles. A flexible workflow and the provision of user-controlled scaffolding and direct manipulation foster a sense of personal control. Interaction authoring tasks are deconstructed for atomic simplicity, so that actions do not have a required order and can easily be added or removed. Immediate feedback and the ability to prototype early are provided with the 3D Window and representations. This fosters exploration and control. Multiple representations and authoring interfaces are provided for working on the design. The provision of multiple, overlapping views helps designers to make connections between parts of the system and tasks, and gain deeper knowledge through reflection.

11.2.3. Initial evaluation of VRBridge

We conducted an exploratory empirical evaluation of VRBridge, once we had created the basic system and the representations. Our aims were as follows:

- To evaluate how well users worked with the representations;
- To evaluate how easily users understood and worked with VRBridge and the Triggersets;
- To determine the scaffolding that would be most effective in helping users to create interactions.

We created two sets of interactions for participants to analyse: in the first they identified interaction sequences; and in the second they identified errors. The interactions included typical VE tasks such as player input triggering an effect, triggers requiring player movement and orientation, and actions triggered by previous interactions, e.g., sounds, animations or movement of VE objects.

We divided participants into two groups, one of which received representations. We hypothesised that participants with representations would produce more accurate solutions in both tasks. Our results confirmed that those with representations consistently out-performed those without: their sequences were a third more accurate and they detected over twice as many errors. They also understood and worked well with the representations. All participants understood the Triggersets and used them to reconstruct interactions.

11.2.4. Designing the final evaluation of VRBridge

Our final evaluation was designed to evaluate the entire VRBridge system, including both design aids. We designed benchmark interactions and tasks, which covered important VR actions, such as object movement across a space avoiding obstacles, rotation on various axes and usage of properties (e.g., sounds). The benchmark interactions were:

- Object reactions to player interaction;
- Sequences of events; and
- Narrative split based on player actions.

We defined three measures for evaluating VR authoring: task performance; computer logging and questionnaires. The task performance measure was the most complex and informative, designed to elicit information about the expertise with which participants worked on the tasks and explored VRBridge and the domain. It consisted of the following factors:

- Successful completion of task requirements.
- Variable Sequencing, measuring effective composition and exploration of Triggersets to create non-linear interactions.
- Inevitable Sequencing, measuring effective usage and exploration of Timelines to create synchronised interactions.
- Player Freedom, measuring player freedom of movement and appropriate inclusion in interactions.

- Creativity, measuring expert creation of interactions, in terms of creating an interesting narrative, effective usage of VE space and natural looking interactions.
- Features, measuring effective usage of system features in the creation of interactions.

For computer logging, we recorded actions of participants with VRBridge so that we could analyse their usage. We also used five questionnaires, four of which we designed and validated ourselves, to elicit results on participant satisfaction. These three measures were triangulated to elicit complex and nuanced evidence about the effectiveness of VRBridge.

We divided participants into four groups so that we could test the effects of our design aids and the success of VRBridge. These were:

- Control group, which received the basic system.
- Representation group, which received the basic system and a Floorplan, Sequence Diagram and Run Mode.
- Scaffolding group, which received the basic system and Tooltips, Context Sensitive Hints and Wizards.
- Combination group, which received the basic system, the representations and the scaffolding.

All participants performed the same three design tasks, which involved creating the interactions for a VE. The tasks increased in complexity, beginning with a learning task where actions were synchronised with sounds, followed by a task which involved moving objects around the VE space, and then a task which required using player interaction to split the narrative. The first two tasks took place in a first session with VRBridge, and third took place a week later.

11.2.5. Results of the final evaluation

Our hypotheses for this evaluation were as follows:

1. *Both representations and scaffolding will improve participant performance. Representations will improve task performance, effective exploration and satisfaction more and scaffolding will improve learning more. The combination of design aids will improve performance the most.* – We found that both representations and scaffolding improved performance according to all four metrics. Representations did not improve task performance more than scaffolding: they were equivalent. However, they did improve exploration (except initially) and satisfaction. Scaffolding did not improve learning more than representations: they were equivalent. The combination of design aids did not improve performance more than either representations or scaffolding, except on satisfaction with learning.
2. *All groups will complete the task more accurately and skilfully in the second session and be more satisfied with the experience.* – We found that all groups completed the task better and were more satisfied with the experience in the second session.

3. *Participants will use the scaffolding less during the second session as they learn about the system and the domain. The scaffolding will be faded.* – We found that the scaffolding was not faded as we thought it would be: the Wizards were not faded and the Context Sensitive Hints were partially faded. Surprisingly, the Tooltips were not turned off; however, they could have been faded through being ignored. The Tooltips require the least attention, offer the least interaction, support the least reflection and are the least sophisticated form of scaffolding. At the other end of the scale, the Wizards require the most attention, offer the most interaction, support the most reflection and are the most sophisticated form of scaffolding. Participants were most satisfied with the Wizards. More sophisticated forms of scaffolding that are task-based will take longer to fade and may never be faded, but offer greater rewards. We showed that, although the Wizards were not faded, expertise in usage increased as participants customised their output more effectively. Therefore, a guideline for provision of more sophisticated scaffolding is that care must be taken to ensure that expertise can continue to grow and continued usage does not hinder the learning process.
4. *We were also interested in how the additional representations were used, both compared to each other and across sessions. Based on our previous study, we expected that the Floorplan would be most used* – Our evaluation showed that all representations were useful: the Floorplan for an overview of the space and using it effectively; the Sequence Diagram for an overview of the interactions, player impact on the interactions and using Triggersets effectively; and the Run Mode for an overview of how the system connects to the running VE and consideration of how the player affects the action. The Floorplan was most used and participants were most satisfied with it. The Sequence diagram and Run Mode require significant attention to achieve design benefits, compared to the Floorplan, which is more obviously and immediately useful. Properties of the representations indicate why satisfaction and usage differed. The Floorplan concretises the VE space, while the Sequence Diagram and Run Mode help the user to think more abstractly about the interactions. The Floorplan allows customisation of the space through direct manipulation and therefore is interactive and makes authoring work easier. In contrast, the Sequence Diagram and Run Mode problematise authoring work by fostering reflection and are less interactive. The Floorplan requires little attention compared to the Sequence Diagram and Run Mode. Therefore, usage of the Sequence Diagram and Run Mode must be supported more carefully, so that they are used more effectively by novices.

11.3. Summary of Evidence for VRBridge as an Authoring Tool

After completing our research, designing VRBridge and the design aids, and evaluating them, we examined evidence for the success of our tool. We consider that we have created an effective authoring tool for end-users if we have accomplished two general sets of aims:

- Providing a tool that is easy and fun to use and learn, and has sufficient power to express VE interactions. If end-users are to work with the tool and make creative products, we must lessen the costs of learning to use the tool and increase the benefits. Lessening the costs is accomplished by lessening learning frustration, increasing task success, ensuring that users can realise their creative designs satisfactorily and easily, and making them want to use the tool.
- Providing a tool that embodies and teaches domain expertise. This is partially accomplished by providing sufficient power to perform domain actions. However, to this we must add an increased understanding of domain concepts, challenges, and requirements for success.

Our evaluations show that VRBridge accomplishes these aims. The basic system provides simple authoring interfaces that our evaluations showed end-users could understand and use immediately to create and analyse interactions. These are flexible enough that end-users work in a variety of ways to achieve their goals. End-users also increased their success and worked in more expert ways within the domain in the later session. After our second evaluation, many participants remarked positively about VRBridge and were pleased and surprised with what they had managed to achieve.

The design aids add considerable benefits to VRBridge. Both representations and scaffolding individually and together significantly improve performance, learning and satisfaction. The effectiveness of the combination of representations and scaffolding on increased satisfaction with learning is important, as end-users who are more satisfied with their learning will be more likely to continue working with the tool and domain. However, design aids must be added carefully, since creativity and exploration are negatively affected as the complexity of the tool increases. This may fade over time; however for tool and domain uptake, early experiences are of primary importance.

The increased satisfaction and usage of the Floorplan and Wizards highlight the importance of interaction and the ability to perform useful work while learning. The representations add enjoyment value and context to the authoring work. We also found that scaffolding improves initial tool experiences, and representations have more long-term effects. End-users with the combination appreciate the benefits of the scaffolding immediately, probably because they see its benefits while using the representations. This shows that adding visual referents to the scaffolding and integrating it more tightly with the authoring work increases its value.

Our evaluations proved that we successfully addressed the problems we found in the literature and our own study of novices. End-users working with VRBridge were able to author non-linearly, understand interaction programming concepts, and consider details while improving the creativity and expertise of their work.

11.4. A Theory-Based Design Method for Authoring Tools

Here, we describe evidence for the success of our theory-based design method for authoring tools. The most compelling evidence is that participants in our experiments were able to use an unfamiliar authoring tool to create several engaging and effective artefacts in less than two hours. The steps of our design method are as follows:

1. Distil practical values from the theory to guide design.
2. Examine processes of expert designers and experts in the domain and connect them to the design values.
3. Conduct research into the practices of novices in the domain to understand their needs and problems.
4. Create a prototype and show it to the target audience. Let them use it. Focus on the design values and their effects in questions afterwards. Then incorporate what has been learned into the artefact. Repeat this step until goal is met.

We chose constructivism as the theory in which to ground our design method. We distilled its key values and then examined expert research in the domain of interest to support our design values and understand how they have been used practically by successful designers. We also examined existing research on novices for guidelines about their problems and typical support. For specific and deeper understanding, we conducted our own study of novices. Thereafter, we created a prototype, focussing on practical implementation of our design values. We designed in two iterations where we evaluated the prototype with users.

Much of the success of our method comes from the choice of constructivism. As we showed above, it supported increased expertise in the novice user. However, other theories may be more suited to different design aims and tools. The value of our method is that it provides a design constraint. When one is creating a tool for others to use, it is easy to be distracted from design values by technological requirements and constraints, and by the discovery of features that add more functionality. By following our theory-based method, the designer is constantly brought back to the underlying theory. Each step requires reconsideration of the theory's principles and how they are being applied. The method also requires that the designer consider the world in which the tool will be used, by considering both how current practitioners do work in the domain and how novices can best be supported in beginning work in the domain. This method therefore delivers a tool which is grounded in appropriate and supported theory, and which is likely to be useful and used within the domain for which it is intended.

11.5. Guidelines for the Design of End-User Authoring Tools and Interactive Systems

Based on our research and experiences, we defined guidelines for the design of end-user authoring tools. These are as follows:

1. Provide simple domain-specific but commonly used visual representations to guide the end-user.
2. Allow practical interaction with the representations and other parts of the interface.
3. Create an attractive environment which will inspire initial design ideas and increase enjoyment.
4. Enable the end-user to create a product as quickly as possible, for positive feedback.
5. Provide integrated multiple forms of support carefully, so as not to make the interface too complex.
6. Where focussed attention is required, scaffold the process through interactive examples and highlighting.
7. Provide a multi-functional tool which helps users to learn, organise their knowledge, work practically with it and share it.

We also defined guidelines for designing interactive systems. These are as follows:

1. Provide a simple authoring interface that highlights important technological considerations.
2. Allow designers to demonstrate, rather than describe their interactions through prototypes.
3. Provide flexibility for how the design work will be accomplished.
4. Provide multiple, linked representations, which are appropriate to the design task. These should at least include an overview of the design space to provide context, and an overview of the interactions, how they connect and how the end-user influences them.
5. Representations should be connected dynamically so that change over time can be viewed.
6. Provide a few simple building blocks which can be combined flexibly and powerfully, rather more complex predefined options. Allow for reuse of designer combinations.
7. Where more knowledge support must be given, it should be tightly integrated with the working environment, interactive and provide useful output.

11.6. Contributions

This thesis provides several novel contributions.

11.6.1. Design method

This thesis describes a theory-based design method for creating end-user systems. The method structures design according to a selected theory: first, practical design values are distilled; then processes of experts and novices in the domain in which the system will be used are researched and the results are considered in terms of the theory-based design values; this is followed by an iterative step where a prototype is created, shown to the target audience and feedback is considered in terms of the design values, the tool is changed and a new prototype is created. While this method is similar to user-centred design methods, it places more emphasis on supporting the development of domain expertise in novices. We designed and followed the method during the course of the thesis, and evaluated its success.

11.6.2. Case study in the use of constructivism to structure a design process

Our design method was based on the constructivist theory of knowledge building. We distilled five design values from constructivism: simplicity, multiplicity, active exploration, reflection, and user control. None of the principles are novel; they have all been used successfully in existing design processes. Our contribution is their consistent application on multiple levels, supported and structured by the underlying theory of constructivism. We used them to define two design aids: multiple, interactive, synchronised domain-specific representations and multiple forms of user-adaptable scaffolding. During the course of the design process we constantly returned to these values and therefore they underpin our method and the resulting tools. Our evaluations showed we had created an effective authoring tool for end-users.

11.6.3. VRBridge

We have created a new authoring tool for end-users designing VR interactions, VRBridge, which is a proof-of-concept prototype of our design method and the authoring aids that resulted. VRBridge provides accessible features for non-programmers and novices in design to easily create 3D interactions. It is modular, extensible, flexible, and allows immediate prototyping. The complementary authoring interfaces of Triggersets and Timelines foster non-linear authoring and skilful synchronisation of interactions without traditional programming. The Triggersets allow complex, non-linear interactions to be flexibly composed from simple building blocks. Spatial markers, waypoints and locations, are provided for working with the space effectively. Two design aids provided with VRBridge, representations and scaffolding, add considerable value to the tool. VRBridge fosters easy learning and usage of the software, and learning about the complexities of the domain so that expertise can be developed. The authoring style, representations and scaffolding provide hints about expert domain practices and foster more expert-like work.

11.6.4. An investigation into effective usage of external representations

The idea of using multiple, interactive, synchronised representations emerged from our analysis of constructivism. External representations themselves form a large research area with many questions about how, where and when they are best used. Our research provides insights and evidence about using representations in software design for end-users, which contribute to this body of literature. Our research began with a review of relevant literature. Thereafter, we conducted our own study of how novices use various kinds of representations. The results of these investigations were applied to our design of representations for VRBridge. Finally, we evaluated our system, gathering results on usage of representations and how they assisted the design process. Our results showed that using multiple representations to decompose, understand and debug VE interactions significantly improves end-user performance and satisfaction.

We confirmed the usefulness of floorplans for providing spatial overviews in 3D and we showed that interactivity adds considerable benefits to representations in terms of satisfaction and uptake. We also showed the value of network diagrams presenting flow of interactions and the impact of the player in interactive systems. Our Sequence Diagram was created by applying simple AI techniques to automatically generate a representation from specified interactions; this close coupling between the representation and the designer's specification provided valuable reflection and practical debugging rewards.

Multiple representations were the most successful design aid that we investigated and showed significant benefits compared to scaffolding and especially compared to the combination.

11.6.5. An investigation into effective usage of scaffolding

Like representations, scaffolding is a large research area, usually connected to constructivist learning. We showed that it can be applied to general learning and not just in a classroom environment. We provide insights into how scaffolding can be used without a human instructor, and insights into the process of fading and its subtleties. We reviewed relevant literature, conducted a study into the particular problems that we should address and applied the results to the scaffolding designed for VRBridge. The results of our evaluations showed that using scaffolding to support authoring improves end-user performance and satisfaction significantly. However, care must be taken that the user is not overwhelmed by adding design aids. In many cases, adding scaffolding to representations lessened their effectiveness as design aids.

We showed that interactivity and visual elements add large benefits to scaffolding, and that more sophisticated, task-based scaffolding is less likely to be faded. We produced the recommendation that these forms of scaffolding should be designed so that end-users can continue to develop expertise even if they do not fade the scaffolding.

11.6.6. Classroom observation of the problems that novices face in programming and design

We conducted an observational study into the problems the problems that novices face when designing interactions and dealing with unfamiliar programming concepts. This provided valuable evidence about how novices work in VR and 3D games, and insights into their problems and successes. The greatest difficulties that novices face in authoring interactions are creating non-linear interactions with alternatives for the player, specifying all the details required to accurately implement interactions, and drawing accurate representations to support their designs.

11.6.7. Empirical user studies and performance metric

To determine the effectiveness of our design method and the resulting tool, we conducted two experiments with end-users, the second of which was a large-scale (124 participants) longitudinal study. These studies

provided empirical evidence about the relative effectiveness of representations and scaffolding as support for designing interactions, about the success of VRBridge and our Triggerset formalism for designing and implementing creative and effective non-linear interactions, and evidence about how representations, scaffolding and the authoring interfaces were used during the authoring process. For the second evaluation, we developed a metric to evaluate the expertise and creativity with which participants author interactions, which can be applied in similar empirical studies.

11.6.8. Guidelines for interactive design systems and systems to support end-user authoring

Based on our extensive research, we provided general guidelines for successfully designing interactive systems and systems to support end-user creative work. These are listed in Section 11.5 above.

11.7. Future Work

There are many further interesting areas of development for this research in the future. A few examples are as follows:

1. VRBridge is a proof-of-concept prototype which addresses interaction authoring in VR. Future work could add to the system, as it is easily extensible. Possible additions include: the ability to generate and manipulate content more effectively; increasing the complexity of the interactions that can be programmed by introducing more atomic actions and triggers; and adding advanced features such as a physics engine, heads up display and the ability to associate or group objects. In addition, the introduction of a method of dealing with uncertainty (such as percentage chances that actions will happen) will increase the expressive power of the system. Embedded time-based actions should also be considered as an alternative or addition to Timelines, to allow non-linear time-based sequences to be created. With these additions, and the ability to generate and add content, VRBridge could become more than a learning tool for novices, and be usable and useful to experts.
2. Following on from the above point, we can see that the event-condition-action system must be extended so that it has the necessary power of expression for more expert designers. At the heart of VR is the player and how the world and objects respond to her; therefore a VE is a reactive system to some degree, making the event-action model a good base for implementing interactions. In future work, we will carefully add to the system, so that it retains the simplicity and logic of the event-action model, but is capable of expressing more complex interactions as described above.
3. Additional representations should be designed and implemented, which focus on different aspects of VR interaction design. The Sequence Diagram considers possible narrative paths through the VE, but is complex to use and focuses on dynamic interaction. Representations could be developed which support more in-depth consideration of narrative potential. For example, a representation which focuses on events in each VE location and how they connect; a representation which promotes consideration of the

character and back story of objects in the VE; or a representation using Fencott's perceptual maps to foster consideration of how the player experiences the story of the VE. In Chapter 3, we made some connections with semiotics, in our discussion of sensory and arbitrary symbols. Further investigation of the field of semiotics could yield ideas for effective representations which highlight the symbolic nature of the design VE.

4. We have shown that VRBridge promotes the creation of VR interactions more expertly. The provision of design aids helps the designer to consider the interactions that are created, not only in the sense of creating a dynamic environment, but also in the sense of creating a meaningful environment. Further reflection on the power of abstraction and the tension between content development and action description would be valuable.
5. Further studies to test the effectiveness of VRBridge include: increasing the complexity and number of interactions and investigating how this influences usage of Sequence Diagram and Run Mode; conducting more long-term studies to investigate fading of scaffolding; and empirically investigating the effectiveness of each form of representation and scaffolding.
6. Another method of evaluation would be to compare VRBridge with other 3D authoring systems, for results about their relative usefulness. Since our focus in this research was on effective design aids and principles which could be applied to any system, we examined VRBridge without comparison to other tools and focussed on the usefulness of our features. We measure the success of VRBridge by users' ability to design interesting and complex interactions in a short amount of time.
7. We believe that many of our results about the effectiveness of representations and scaffolding are also applicable to more expert users, both domain experts learning new software and programmers exploring design. Studies could be conducted which evaluate the usefulness of our representations and scaffolding for different kinds of users.
8. Our guidelines for designing interactive systems and end-user authoring systems, and our theory-based design method can be systematically applied to design of other systems, and therefore tested practically.

11.8. Closing Remarks

This thesis set out to investigate how best to support end-users in authoring interactive systems, to demonstrate that authoring systems can be made more effective by following a theory-based design method, and to investigate how constructivism could be used to support design practically. We achieved all of these goals: we created an effective authoring system in VRBridge, which was thoroughly empirically validated; we documented our design method and showed how each step contributed to the success of the final system; and we showed how constructivism could be used outside a traditional learning environment to support learning a tool and how to work in a domain.

We also showed that providing design support in the form of representations or scaffolding significantly improves end-user performance and satisfaction. This is a key finding in helping to make VR more accessible as a creative medium. Our research was comprehensive enough to provide general guidelines for effectively designing end-user authoring and interactive systems.

VRBridge creates a space for end-users to explore design possibilities and be inspired by them. The techniques and support for more expert work are embedded in the tool and provide resources and space for creativity to emerge. Our most compelling evidence is that, with very little experience in 3D authoring and using a new tool, end-users were able to author creative and interesting complex interactions within one hour of working with VRBridge, and their expertise increased measurably during only two authoring sessions. This provides valuable insights into how to create systems which help and inspire end-users to do practical creative work, achieve satisfying results and produce effective artefacts.

References

Adelson, B. 1981. Problem solving and the development of abstract categories in programming languages. *Memory and Cognition*, 9: 422-433.

AgentSheets Reference Manual [Online]. Available: http://agentsheets.com/ [25 January 2010]

Allison, D., Wills, B., Bowman, D., Wineman, J. & Hodges, L. 1997. The virtual reality gorilla exhibit. *IEEE Computer Graphics and Applications*, 17: 30-38.

Anastasi, A. & Urbina, S. 1996. *Psychological Testing*. New York: Prentice Hall.

Azevado, R., Cromley, J. & Siebert, D. 2004. Does adaptive scaffolding facilitate students' ability to regulate their learning with hypermedia? *Contemporary Educational Psychology,* 29: 344-370.

Azevado, R. & Hadwin, A. 2005. Scaffolding self-regulated learning and metacognition – implications for the design of computer-based scaffolds. *Instructional Science*, 33: 367-379.

Baldonado, M., Woodruff, A. & Kuchinsky, A. 2000. Guidelines for using multiple views in information visualization. *Proceedings of the Workshop on Advanced Visual Interfaces*, Palermo, 23-26 May 2000. New York: ACM Press.

Banister, P., Burman, E., Parker, I., Taylor, M. & Tindall, E. 1994. *Qualitative Methods in Psychology: A Research Guide*. Buckingham: Open University Press.

Battista, G.D., Eades, P., Tamassia, R. & Tollis, I.G. 1994. Algorithms for drawing graphs: an annotated bibliography. *Computational Geometry: Theory and Applications*, 4: 235–282.

Beirowski, C. & Vermuelen, H. 2004. Experiences with virtual reality accessibility in an African context. *Workshop proceedings of the IEEE VR 2004 Conference*, Chicago, pp. 14-18.

Blackwell, A.F. 1997. Diagrams about thoughts about thoughts about diagrams. *Reasoning with Diagrammatic Representations II: Papers from the Association for the Advancement of Artificial Intelligence 1997 Symposium*, Fall, pp. 77–84.

Blackwell, A.F. 2001a. See what you need: helping end-users to build abstractions. *Journal of Visual Languages and Computing*, 12: 475-499.

Blackwell, A.F. 2001b. Pictorial representation and metaphor in visual language design. *Journal of Visual Languages and Computing*, 12:223–252

Blackwell, A.F. 2001c. First steps in programming: a rationale for attention investment models. *Proceedings of the IEEE Symposia on Human-Centric Computing Languages and Environments*, Virginia, USA, 3-6 September 2002, pp. 2-10.

Blackwell, A.F. 2002. What is programming? *Proceedings of the 14th Workshop of the Psychology of Programming Interest Group*, Brunel University, June, pp. 204-218.

Blackwell, A.F. & Engelhardt, Y. 2002. A meta-taxonomy for diagram research. (*In*: Olivier, P., Anderson, M. & Meyer, B., (eds.) *Diagrammatic Representation and Reasoning*. Springer-Verlag.)

Blackwell, A.F., Whitley, K.N., Good, J. & Petre, M. 2001. Cognitive factors in programming with diagrams. *Artificial Intelligence Review*, 15(1): 95-113.

Bowman, D.A. & Hodges, L. 1999. Formalizing the design, evaluation and application of interaction techniques for immersive virtual environments. *Journal of Visual Languages and Computing, 10*: 37–53.

Bowman, D.A., Kruijff, E., LaViola, J.J. & Poupyrev, I. 2001. An introduction to 3D user interface design. *Presence: Teleoperators and Virtual Environments*, 10(1): 96-108.

Bowman, D.A., Gabbard, J. & Hix, D. 2002. A Survey of usability evaluation in virtual environments: classification and comparison of methods. *Presence: Teleoperators and Virtual Environments*, 11(4): 404-424.

Brown, S., Ladeira, I., Winterbottom, C. & Blake, E. 2003. The effects of mediation in a storytelling virtual environment. (*In*: Balet, O. Subsol, G. & Torguet, P., (eds.) *Lecture Notes in Computer Science, vol. 2897*. Springer-Verlag. p. 102-111.)

Brown, S., Nunez, D. & Blake, E. 2005 Using virtual reality to provide nutritional support to HIV+ women. *Proceeding of the 8th International Workshop on Presence*, University College, London, pp. 319-326.

Bruner, J. 1966. *Toward a Theory of Instruction*. Harvard University Press.

Buckingham, D. & Burn, A. 2007. Game literacy in theory and practice. *Journal of Educational Multimedia and Hypermedia*, 16(3): 323-349.

Burnett, M. 2009. What is end-user software engineering and why does it matter? (*In*: Pipek, V., Rosson, M.B., Ruyter, B. & Wulf, V. (eds.) *Lecture Notes In Computer Science, vol. 5435*. Springer-Verlag, p. 15-28.)

Carassa, A., Morganti, F. & Tirassa, M. 2004. Movement, action, and situation: presence in virtual environments. *Proceedings of the 7th Annual International Workshop on Presence*, Valencia, Spain, 13-15 October 2004, pp. 7-12.

Card, S.K., Mackinlay, J.D. & Shneiderman, B. (eds.). 1999. *Readings in Information Visualization: Using Vision to Think*. California: Morgan Kaufmann Publishers.

Chase, W.G. & Simon, H.A. 1973. The mind's eye in chess. (*In*: Chase, W.G. (ed.). *Visual Information Processing*. New York: Academic Press. p. 215-281.)

Chen, X. & Kalay, Y. 2008. Making a liveable 'place': content design in virtual environments. *International Journal of Heritage Studies*, 14(3): 229 - 246.

Cheng, P., Lowe, R. & Scaife, M. 2001. Cognitive science approaches to understanding diagrammatic representations. *Artificial Intelligence Review*, 15(1): 79-94.

Chi, M.T. 2006. Two approaches to the study of experts' characteristics. (*In*: Ericsson, K.A, Charness, N., Feltovich, P.J. & Hoffman, R.R. (eds.) *The Cambridge Handbook of Expertise and Expert Performance*. Cambridge University Press, p. 21-30.)

Chin, J. P., Diehl, V. A. & Norman, K. L. 1988. Development of an instrument measuring user satisfaction of the human-computer interface. *Proceedings of the Special Interest Group on Computer-Human Interaction, SIGCHI 1988*, pp. 213-218.

Church, D. 1999. Formal abstract design tools. Game Developer Magazine, August 1999. [Online]. Available: http://www.gamasutra.com/features/19990716/design_tools_01.htm [25 January 2010]

Conway, M. 1997. *Alice: easy-to-learn 3D scripting for novices*. Charlottesville, USA. School of Engineering and Applied Science, University of Virginia. (PhD Thesis.)

Conway, M., Audia, S., Burnette, T., Cosgrove, D., Christiansen, K., Deline, R., Durbin, J., Gossweiler, R., Koga, S., Long, C., Mallory, B., Miale, S., Monkaitis, K., Patten, J., Pierce, J., Shochet, J., Staack, D., Stearns, B., Stoakley, R., Sturgill, C., Viega, J.,White, J., Williams, G., & Pausch, R. 2000. Alice: lessons learned from building a 3D system for novices. *Proceedings of the ACM CHI 2000 Human Factors in Computer Systems Conference*, The Hague, Netherlands, 1-6 April 2000, pp. 486-493.

Cooper, S., Dann, W. & Pausch, R. 2003. Teaching objects-first in introductory computer science. *Proceedings of 34th SIGCSE Technical Symposium on Computer Science Education*, Reno, USA, 19-23 February 2003, pp. 191-195.

Crawford, C. 1984. *The Art of Computer Game Design*. McGraw-Hill Publishers.

Crismond, D. 2001. Learning and using science ideas when doing investigate-and-redesign tasks: a study of naive, novice, and expert designers doing constrained and scaffolded design work. *Journal of Research in Science Teaching*, 38(7): 791-820.

Cronbach, L. J. 1990. *Essentials of Psychological Testing*. New York: Addison Wesley.

Cross, N. 2001. Designerly ways of knowing: design discipline versus design science. *Design Issues*, 17(3): 49-55. MIT Press.

Cross, N. 2004. Expertise in design: an overview. *Design Studies*, 25(5): 427-441.

Darken, R. & Peterson, B. 2002. Spatial orientation, wayfinding and representation. (*In*: Stanney, K. (ed.) *Handbook of Virtual Environments: Design, Implementation, and Applications*. Lawrence Erlbaum Associates, p. 493-518.)

Do, E. & Gross, M.D. 2001. Thinking with diagrams in architectural design. *Artificial Intelligence Review*, 15: 135-149.

Dorst, K. 2006. Design problems and design paradoxes. *Design Issues*, 22(3): 4-17. MIT Press.

Dow, S., Saponas, T.S., Li, Y. & Landay, J.A. 2006. External representations in ubiquitous computing design and the implications for design tools. *Proceedings of the Conference on Designing Interactive Systems '06*, University Park, PA, USA, 26-28 June 2006, pp. 241-250. ACM Press.

Eastman, C. 1999. Representation of design processes. *Invited Keynote Address at Conference on Design Thinking*, 23-25 April 1999.

Eastman, C. 2000. New directions in design cognition: studies of representation and recall. (*In*: Eastman, C., McCracken, W.M. & Newstetter, W. (eds.) *Design Knowing and Learning: Cognition in Design Education*. Elsevier Press, p. 147-198.)

Ericsson, K.A. 2006. An introduction to the Cambridge Handbook of Expertise and Expert Performance: its development, organization, and content. (*In*: Ericsson, K.A, Charness, N., Feltovich, P.J. & Hoffman, R.R. (eds.) *The Cambridge Handbook of Expertise and Expert Performance*. Cambridge University Press, p. 3-20.)

Ericsson, K.A. & Charness, N. 1994. Expert performance: its structure and acquisition. *American Psychologist*, 49(8): 725-747.

Ericsson, K.A., Krampe, R. & Tesch-Romer, C. 1993. The role of deliberate practice in the acquisition of expert performance. *Psychological Review*, 100(3): 363-406.

Feltovich, P.J., Prietula, M. & Ericsson, K.A. 2006. Studies of expertise from psychological perspectives. (*In*: Ericsson, K.A, Charness, N., Feltovich, P.J. & Hoffman, R.R. (eds.) *The Cambridge Handbook of Expertise and Expert Performance*. Cambridge University Press, p. 41-60.)

Fencott, C. 1999. Content and creativity in virtual environment design. *Proceedings of Virtual Systems and Multimedia '99*, University of Abertay, Dundee, Scotland.

Fencott, C. 2001a. Comparative content analysis of virtual environments using perceptual opportunities. (*In*: Earnshaw, R. & Vince, J. (eds.) *Digital Content Creation*. Springer-Verlag, p. 25-51.)

Fencott, C. 2001b. Virtual storytelling as narrative potential: towards an ecology of narrative. (In: Balet, O. Subsol, G. & Torguet, P., (eds.) *Lecture Notes in Computer Science*, vol. 2197. Springer-Verlag. p. 90–99.)

Fosnot, C.T. 1996. Constructivism: a psychological theory of learning. (*In*: Fosnot, C.T. (ed.) *Constructivism: Theory, Perspectives and Practice*. Teachers College Press, p. 8-33.)

Freeman, J., & Avons, S. E. 2000. Focus group exploration of presence through advanced broadcast services. *Proceedings of the SPIE, Human Vision and Electronic Imaging Conference V*, pp. 3959-3976.

Fullerton, T., Chen, J., Santiago, K., Nelson, E., Diamante, V., & Meyers, A. 2006. That Cloud game: dreaming (and doing) innovative game design. *Proceedings of the 2006 ACM SIGGRAPH Symposium on Videogames*, Boston, Massachusetts, 30-31 July 2006, pp.51-59. ACM Press.

Gibson, J.J. 1971. The information available in pictures. *Leonardo*, 4: 27-35. Pergammon Press.

Gibson, J.J. 1979. *The Ecological Approach to Visual Perception*. Houghton Mifflin Company.

Gordon, I.E. 2004. *Theories of Visual Perception*. 3rd ed. Psychology Press.

Gregory, R.L. 1997. Knowledge in perception and illusion. *Philosophical Transactions of the Royal Society of London: Biological Sciences*, 352: 1121-1128.

Gross, D. 2002. Technology management and user acceptance of virtual environment technology. (*In*: Stanney, K. (ed.) *Handbook of Virtual Environments: Design, Implementation, and Applications.* Lawrence Erlbaum Associates, p. 533-542.)

Guindon, R., Krasner, H. & Curtis, B. 1987. Breakdowns and processes during the early activities of software design by professionals. (*In*: Soloway, E. & Sheppard, S. (eds.) *Empirical Studies of Programmers: Second Workshop*. Intellect Books, p. 65-82.)

Guzdial, M. 1994. Software-realized scaffolding to facilitate programming for science learning. *Interactive Learning Environments*, 4(1): 1-44.

Harada, K., Tanaka, E., Ogawa, R. & Hara, Y. 1996. Anecdote: a multimedia storyboarding system with seamless authoring support. *Proceedings of the Fourth ACM international Conference on Multimedia*, Boston, Massachusetts, 18-22 November 1996, pp. 341–351.

Harel, D. 1987. Statecharts: a visual formalism for complex systems. *Science of Computer Programming*, 8: 231-274.

Hendricks, Z., Tangkuampien, J. & Malan, K. 2003. Virtual galleries. Is 3D better? *Proceedings of the 2nd International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa (AFRIGRAPH)*, Cape Town, South Africa, 3-5 February 2003, pp. 17-24.

Herman, I., Melancon, G. & Marshall, M.S. 2000. Graph visualisation and navigation in information visualisation: a survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1): 24-43.

Hewett, T. 2005. Informing the design of computer-based environments to support creativity. *International Journal of Human-Computer Studies*, 63: 383-409. Elsevier Ltd.

Hoffman, H.G., Garcia-Palacios, A., Carlin, A., Furness III, T.A. & Botella-Arbona, C. 2003. Interfaces that heal: coupling real and virtual objects to treat spider phobia. *International Journal of Human-Computer Interaction*, 16(2): 283-300.

IJsselsteijn, W. A., de Ridder, H., Freeman, J. & Avons, S. E. 2000. Presence: concept, determinants and measurement. *Proceedings of the SPIE, Human Vision and Electronic Imaging V*, pp. 3959-3976.

Irani, P., Tingley, M. & Ware, C. 2001. Using perceptual syntax to enhance semantic content in diagrams. *IEEE Computer Graphics and Applications*, 21(5): 76-85.

Jackson, S., Krajcik, J. & Soloway, E. 1998. The design of guided learner-adaptable scaffolding in interactive learning environments. *Proceedings of CHI '98*, Los Angeles, California, 18-23 April 1998, pp. 187-194.

Jenkins, H. 2004. Game design as narrative architecture. (*In*: Wardrip-Fruin, N. & Harrigan, P. (eds.) *First Person: New Media as Story, Performance, Game*, MIT Press.)

Jonassen, D. 1999. Constructivist learning environments on the web: engaging students in meaningful learning. *EdTech 99: Educational Technology Conference and Exhibition 1999: Thinking Schools, Learning Nation*, 9-11 February 1999.

Jonassen, D. 2000. Toward a design theory of problem solving. *Educational Technology Research and Development*, 48(4): 63-85.

Jonassen, D. & Carr, C. 2000. Mindtools: affording multiple knowledge representations for learning. (*In*: Lajoie, S.P. (ed.) *Computers as Cognitive Tools, Volume 2: No More Walls*, Lawrence Erlbaum Associates, p. 165-195.)

Kao, M. & Lehman, J. 1997. Scaffolding in a computer-based constructivist environment for teaching statistics to college learners. *Proceedings of Annual Meeting of the American Educational Research Association (AERA)*, March 1997, pp. 1-22.

Kaur, K. 1998. *Designing Virtual Environments for Usability*. London, UK, Centre for Human-Computer Interface Design, City University. (PhD Thesis.)

Kaur, K., Sutcliffe, A. & Maiden, N. 1999. A design advice tool presenting usability guidance for virtual environments. *Proceedings of Workshop on User Centred Design and Implementation of Virtual Environments*, York, England.

Kavakli, M., Suwa, M., Gero, J. & Purcell, T. 1999. Sketching interpretation in novice and expert designers. (*In*: Gero, J. S. and Tversky, B. (eds) *Visual and Spatial Reasoning in Design*. University of Sydney, Australia, p. 209-220.)

Kelleher, C. & Pausch, R. 2005. Lowering the barriers to programming: a taxonomy of programming environments and language for novice programmers. *ACM Computing Surveys*, 37(2): 83-137.

Kissinger, C., Burnett, M., Stumpf, S., Subrahmaniyan, N., Beckwith, L., Yang, S. & Rosson, M. 2006. Supporting end-user debugging: what do users want to know? *Proceedings of the Working Conference on Advanced Visual interfaces*, Venezia, Italy, 23-26 May 2006, pp. 135-142. ACM Press.

Ko, A.J., Myers, B. A. & Aung, H.H. 2004. Six learning barriers in end-user programming systems. *Proceedings of IEEE Symposium of Visual Languages and Human Centric Computing*, Rome, Italy, 26-29 September 2004, pp. 199-206.

Ko, A.J. & Myers, B.A. 2005. A framework and methodology for studying the causes of software errors in programming systems. *Journal of Visual Languages and Computing*, 16: 41-84.

Krisler, B. & Alterman, R. 2008. Training towards mastery: overcoming the active user paradox. *Proceedings of the 5th Nordic Conference on Human-Computer interaction: Building Bridges*, Lund, Sweden, 20-22 October 2008, pp. 239-248. ACM Press.

Ladeira, I. & Blake, E. 2004. Virtual san storytelling for children: content vs. experience. *Proceedings of the 5th International Symposium on Virtual Reality, Archaeology and Cultural Heritage VAST 2004*, Brussels, Belgium, 7-10 December 2004, pp. 223-231. The Eurographics Association.

Larkin, J. & Simon, H. 1987. Why a diagram is (sometimes) worth ten thousand words. *Cognitive Science*, 11: 65-99.

Laurel, B. 1993. *Computers as Theatre*. Addison-Wesley Publishing Company.

Leão, L. 2002. The labyrinth as a model of complexity: the semiotics of hypermedia. *Proceedings of Computational Semiotics for New Media (COSIGN '02)*, 2-4 September 2002.

Lessiter, J., Freeman, J., Keogh, E. & Davidoff, J. 2001. A cross-media presence questionnaire: the ITC-Sense of Presence Inventory. *Presence: Teleoperators and Virtual Environments*, 10(3): 282-297.

Lombard, M. & Ditton, T. 1997. At the heart of it all: the concept of presence. *Journal of Computer-Mediated Communication*, 3(2).

Mirakhorli, M., Khanipour Rad, A., Shams, F., Pazoki, M. & Mirakhorli, A. 2008. RDP technique: a practice to customize XP. *Proceedings of the 2008 International Workshop on Scrutinizing Agile Practices Or Shoot-Out At the Agile Corral*, Leipzig, Germany, 10 May 2008, pp. 23-32.

Murray, J. 1997. *Hamlet on the Holodeck: the Future of Narrative in Cyberspace*. New York: The Free Press.

Myers, B.A. 1990. Taxonomies of visual programming and program visualization. *Journal of Visual Languages and Computing*, 1: 97-123

Myers, B.A., Hudson, S. & Pausch, R. 2000. Past, present and future of user interface software tools. *ACM Transactions on Computer-Human Interaction*, 7 (1): 2-28. ACM Press.

Myers, B.A., Pane, J.F. & Ko, A. 2004. Natural programming languages and environments. *Communications of the ACM*, 47(9): 47-52.

Myers, B.A., Park, S.Y., Nakano, Y., Mueller, G. & Ko, A. 2008. How designers design and program interactive behaviours. *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing*, 15-19 September 2008, pp. 177-184.

Neisser, U. 1994. Multiple systems: a new approach to cognitive theory. *European Journal of Cognitive Psychology*, 6(3): 225–41.

Nielsen, J. & Molich, R. 1990. Heuristic evaluation of user interfaces. *Proceedings of ACM CHI'90 Conference*, Seattle, Washington, 1-5 April 1990, pp. 249-256.

Nielsen, J. 2005. *Ten Usability Heuristics*. [Online]. Available: http://www.useit.com/papers/heuristic/heuristic_list.html [25 January 2010].

Norman, D. 1988. *The psychology of everyday things*. Harper Collins Publishers.

Norman, D. 1993. *Cognition in the head and in the world. Cognitive Science*, 17: 1-6.

Norman, D. 1999. *The Invisible Computer*. MIT Press.

Norman, J. 2002. Two visual systems and two theories of perception: an attempt to reconcile the constructivist and ecological approaches. *Behavioral and Brain Sciences*, 25: 73–144.

O'Brien, M. P. & Buckley, J. 2001. Inference-based and expectation-based processing in program comprehension. *Proceedings of the 9th International Workshop on Program Comprehension*, Toronto, Canada, 12-13 May 2001, p. 71. IEEE Computer Society.

Orey, M. (ed.). 2001. *Emerging Perspectives on Learning, Teaching, and Technology*. [Online]. Available: http://www.coe.uga.edu/epltt/index.htm [25 January 2010].

Palmer, S.E. & Rock, I. 1994. Rethinking perceptual organization: the role of uniform connectedness. *Psychonomic Bulletin and Review*, 1(1): 29-55.

Pane, J. & Myers, B. 1996. Usability Issues in the Design of Novice Programming Systems. Human-Computer Interaction Institute Technical Report. School of Computer Science, Carnegie Mellon University [Online]. Available: www.cs.cmu.edu/~pane/cmu-cs-96-132.html [25 January 2010].

Pane, J., Ratanamahatana, C. & Myers, B. 2001. Studying the language and structure in non-programmers' solutions to programming problems. *International Journal of Human-Computer Studies*, 54: 237-264. Academic Press.

Papert, S. & Harel, I. 1991. Situating Constructionism. (*In*: Papert, S. & Harel, I. (eds.) *Constructionism*. Ablex Publishing Corporation, Chapter 1.)

Pausch, R., Burnette, T., Capehart, A.C., Conway, M., Cosgrove, D., DeLine, R., Durbin, J., Gossweiler, R., Koga, S. & White, J. 1995. A brief architectural overview of Alice, a rapid prototyping system for virtual reality. *IEEE Computer Graphics and Applications*, 15(3): 195-203.

Pearce, C. & Ashmore, C. 2007. Principles of emergent design in online games: Mermaids phase 1 prototype. *Proceedings of the ACM SIGGRAPH Sandbox Symposium '07*, San Diego, California, 4-5 August 2007, pp. 65-71. ACM Press.

Petre, M. & Blackwell, A.F. 1999. Mental imagery in program design and visual programming. *International Journal of Human-Computer Studies*, 51: 7-30.

Piaget, J. 1937. *The Child's Construction of Reality*. [English Translation 1955]. London: Routledge.

Preece, J., Rogers, Y. & Sharp, H. 2002. *Interaction Design: Beyond Human-Computer Interaction*. John Wiley and Sons.

Purchase, H.C. 1997. Which aesthetic has the greatest effect on human understanding? *Proceedings of the 5th Annual Symposium on Graph Drawing*, Rome, Italy, 18-20 September 1997, pp. 248-261.

Quintana, C., Reiser, B., Davis, E., Krajcik, J., Golan, R., Kyza, E., Edelson, D. & Soloway, E. 2002. Evolving a scaffolding design framework for designing educational software. *Proceedings of the 5th International Conference of the Learning Sciences*, Seattle, Washington, 23-26 October 2002.

Reiser, B. 2004. Scaffolding complex learning: the mechanisms of structuring and problematising student work. *The Journal of Learning Sciences*, 13(3): 273-304. Lawrence Erlbaum Associates.

Repenning, A., Ioannidou, A. & Phillips, J. 1999. Collaborative use and design of interactive simulations. *Proceedings of ACM Conference on Computer Supported Cooperative Learning*, Palo Alto, California, 12-15 December 1999. ACM Press.

Repenning, A., Ioannidou, A. & Zola, J. 2000. AgentSheets: end-user programmable simulations. *Journal of Artificial Societies and Social Simulation*, 3(3).

Rock, I. 1997. *Indirect Perception*. Cambridge: MIT Press.

Salen, K. & Zimmerman, E. 2003. *Rules of Play: Game Design Fundamentals*. MIT Press.

Saltzman, M. 2000. *Game Design: Secrets of the Sages*. 2nd ed. Brady Games.

Saltzman, M. 2003. *Game Creation and Careers: Insider Secrets from Industry Experts*. New Riders.

Scaife, M. & Rogers, Y. 1996. External cognition: how do graphical representations work? *International Journal of Human-Computer Studies*, 45: 185-213.

Scaife, M. & Rogers, Y. 2005. External cognition, innovative technologies, and effective learning. (*In*: Gardenfors, P. & Johansson, P. (eds.) *Cognition, Education and Communication Technology*. Lawrence Erlbaum Associates. p. 181-202.)

Schön, D.A. 1983. *The Reflective Practitioner: How Professionals Think in Action*. New York: Harper Collins.

Schön, D.A. 1987. *Educating the Reflective Practitioner: Towards a New Design for Teaching and Learning in the Professions*. Jossey-Bass Inc.

Schön, D.A. & Wiggins, G. 1992. Kinds of seeing and their functions in designing. *Design Studies*, 13(2): 135-156.

Schubert, T., Friedmann, F. & Regenbrecht, H. 1999a. Decomposing the sense of presence: factor analytic insights. *Proceedings of the 2nd International Workshop on Presence*, Colchester, England, 6-7 April 1999.

Schubert, T., Friedmann, F. & Regenbrecht, H. 1999b. Embodied presence in virtual environments. (*In*: Paton, R. & Neilson, I. (eds.) *Visual Representations and Interpretations*. Springer-Verlag, p. 269-278.)

Schuemie, M., Van Der Straaten, P., Krijn, M. & Van Der Mast, C. 2001. Research on presence in virtual reality: a survey. *CyberPsychology & Behaviour*, 4(2): 183–201.

Sheridan, T.B. 1999. Descartes, Heidegger, Gibson, and God: towards an eclectic ontology of presence. *Presence*, 8(5): 551-559.

Sherin, B., Reiser, B. & Edelson, D. 2004. Scaffolding analysis: extending the scaffolding metaphor to learning artifacts. *Journal of the Learning Sciences*, 13(3): 387-421.

Shneiderman, B. 1992. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. 2nd ed. Massachusetts: Addison-Wesley Publishing.

Shneiderman, B. 1996. The eyes have it: a task by data type taxonomy for information visualizations. *Proceedings of the '96 IEEE Symposium on Visual Languages*, September 1996, pp. 336-343.

Shneiderman, B. 2000. Creating creativity: user interfaces for supporting innovation. *ACM Transactions on Computer-Human Interaction*, 7(1): 114-138. ACM Press.

Shotter, J. 1995. Dialogue: social constructionism and radical constructivism. (*In*: Steffe, L.P. & Gale, J. (eds.). *Constructivism in Education*. Lawrence Erlbaum Associates, p. 41–56.)

Shu, N.C. 1988. *Visual Programming*. New York: Van Nostrand Reinhold Company.

Simon, H. A. 1973. The structure of ill structured problems. *Artificial Intelligence*, 4: 181-201.

Slater, M. & Usoh, M. 1993. Representation systems, perceptual positions, and presence in immersive virtual environments. *Presence: Teleoperators and Virtual Environments*, 2(3):221–233.

Slater, M. & Wilbur, S. 1997. A framework for immersive virtual environments (FIVE): Speculations on the role of presence in virtual environments. *Presence*, 6(6): 603–616.

Sobek, D.K. 2002. Representation in design: data from engineering journals. *Proceedings of 32nd ASEE/IEEE Frontiers in Education Conference*, Boston, Massachusetts, 6-9 November 2002.

Soloway, E. 1986. Learning to program = learning to construct mechanisms and explanations. *Communications of the ACM*, 29(9): 850-858.

Spiro, R., Feltovich, P., Jacobson, M. & Coulson, R. 1992. Cognitive flexibility, constructivism, and hypertext: random access instruction for advanced knowledge acquisition in ill-structured domains. (*In*: Duffy, T. & Jonassen, D. (eds.) *Constructivism and the technology of instruction*. Lawrence Erlbaum Associates, p. 57-76.)

Spradley, J. 1998. *Participant Observation*. New York: Holt, Rinehart and Winston.

Stanney, K. (ed.). 2002. *Handbook of Virtual Environments: Design, Implementation, and Applications*, Lawrence Erlbaum Associates.

Subrahmaniyan, N., Burnett, M. & Bogart, C. 2008. Software visualization for end-user programmers: trial period obstacles. *Proceedings of the 4th ACM Symposium on Software Visualization*, Ammersee, Germany, 16-17 September 2008, ACM Press, pp. 135-144.

Tangkuampien, J. 2005. *Virtual Environment Authoring Interface for Content-Expert Authors*. Cape Town, South Africa, Computer Science Department, University of Cape Town. (Masters Thesis).

Tanriverdi, V. & Jacob, R. 2001. VRID: A design model and methodology for developing virtual reality interfaces. *Proceedings of the Virtual Reality Software and Technology Conference*. Banff, Alberta, Canada, 15-17 November 2001, pp. 175-182. ACM Press.

Tognazzini, B. 2003. *First Principles of Interaction Design*. [Online]. Available: http://www.asktog.com/basics/firstPrinciples.html [25 January 2010].

Tory, M., Kirkpatrick, A., Atkins, M. & Moller, T. 2006. Visualization task performance with 2D, 3D and combination displays. *IEEE Transactions on Visualization and Computer Graphics*, 12(1): 2-13.

Tufte, E. 1983. *The Visual Display of Quantitative Information*. Cheshire, UK: Graphics Press.

Tufte, E. 1997. *Visual Explanations: Images and Quantities, Evidence and Narrative*. Cheshire, UK: Graphics Press.

Usoh, M., Arthur, K., Whitton, M., Bastos, R., Steed, A., Slater, M. & Brooks, F. 1999. Walking>walking in place>flying, in virtual environments. *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 359-364.

van Wijk, J. 2005. The value of visualization. *Proceedings of IEEE Visualization 2005*, pp. 79–86.

von Glasersfeld, E. 1995. A constructivist approach to teaching. (*In*: Steffe, L.P. & Gale, J. (eds.). *Constructivism in Education*. Lawrence Erlbaum Associates, p. 3–16.)

von Glasersfeld, E. 1996. Introduction: aspects of constructivism. (*In*: Fosnot, C.T. (ed.) *Constructivism: Theory, Perspectives and Practice*. Teachers College Press, p. 3-7.)

Vygotsky, L.S. 1934. *Thought and Language*. [English Translation 1986] Cambridge, MA: MIT Press.

Vygotsky, L.S., Cole, M. (ed.), John-Steiner, V. (ed.), Scribner, S. (ed.). 1978. *Mind in Society: The Development of Higher Psychological Processes*. Harvard University Press.

Ware, C. 2000. *Information Visualisation: Perception for Design*. Morgan Kaufmann Publishers.

Ware, C. 2001. Designing with a 2.5D attitude. *Information Design Journal*, 10(3): 255-262.

Ware, C., Purchase, H., Colpoys, L. & McGill, M. 2002. Cognitive measurements of graph aesthetics. *Information Visualisation*, 1(2): 103-110.

Wazlawick, R., Rosatelli, M., Ramos, E., Cybis, W., Storb, B., Schuhmacher, V., Mariani, A., Kirner, T., Kirner, C. & Fagundes, L. 2001. Providing more interactivity to virtual museums: a proposal for a VR authoring tool. *Presence*, 10(6): 647-656. MIT Press.

Winterbottom, C. & Blake, E. 2004. Designing a VR interaction authoring tool using constructivist practices. *Proceedings of the 3rd International Conference on Virtual Reality, Computer Graphics, Visualization and Interaction in Africa*, Stellenbosch, South Africa, November 2004, pp. 67-71. ACM Press.

Winterbottom, C. & Blake, E. 2008. Constructivism, virtual reality and tools to support design. *Proceedings of the 7th ACM Conference on Designing Interactive Systems '08*, Cape Town, South Africa, January 2008, pp. 230-239. ACM Press.

Winterbottom, C., Gain, J. & Blake, E. 2006. Using visualizations to support design and debugging in virtual reality. (*In*: Bebis, G. et al. (eds.) *Lecture Notes in Computer Science, vol. 4291: Advances in Visual Computing*. Springer-Verlag. p. 465-474.)

Wolber, D. 1998. A multiple timeline editor for developing multi-threaded animated interfaces. *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology*, San Francisco, California, pp. 117–118.

Wood, D., Bruner, J. & Ross, G. 1976. The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry*, 17: 89-100.

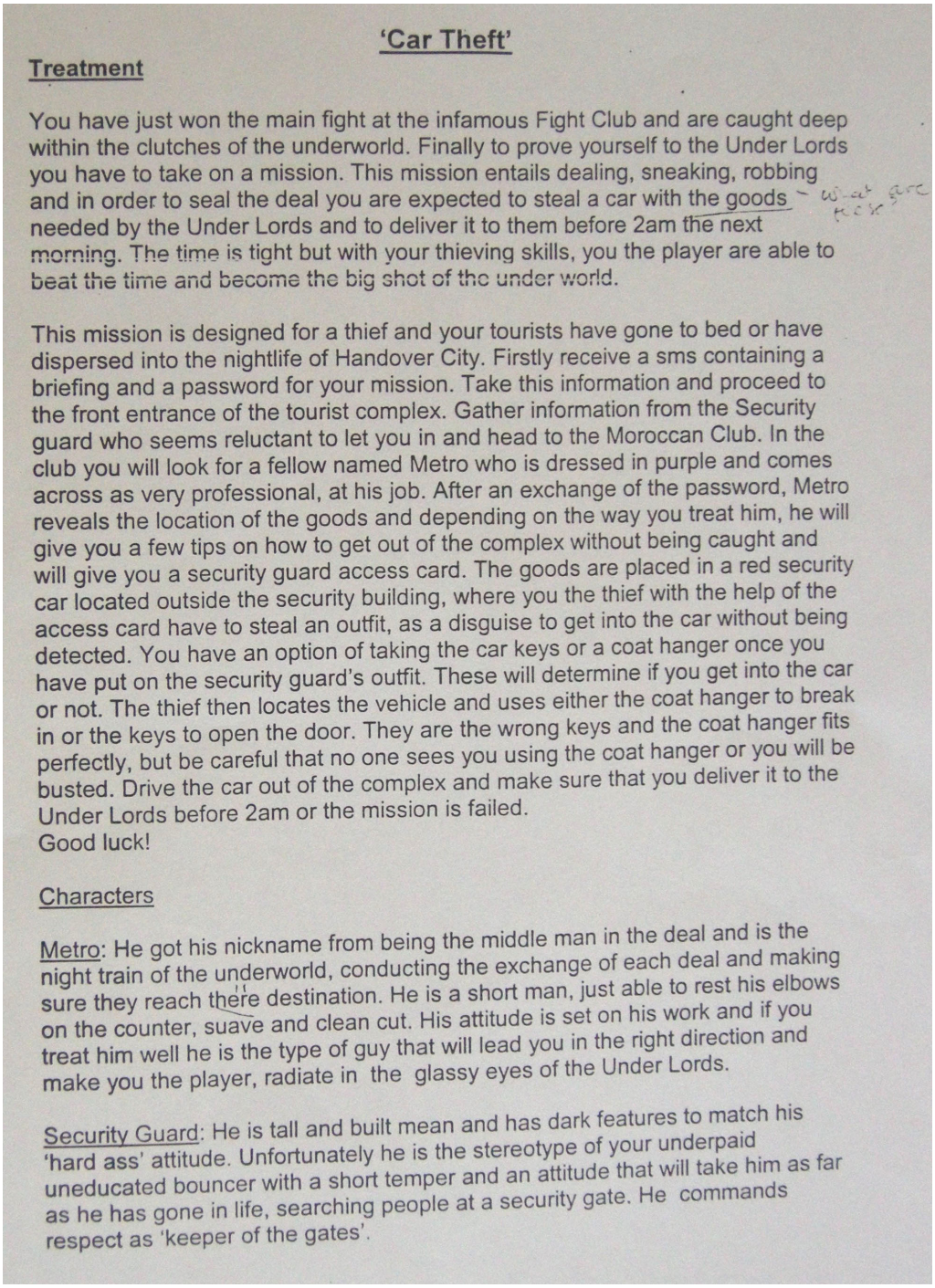
Zachmann, G. 1996. A language for describing behaviour of and interaction with virtual worlds. *Proceedings of ACM Virtual Reality Software and Technology Conference '96*, Hong Kong, July 1996, pp. 143-150.

Zhang, J. & Norman, D. 1994. Representations in distributed cognitive tasks. *Cognitive Science*, 18: 87-122.

Zhang, J. 1997. The nature of external representations in problem solving. *Cognitive Science*, 21: 179-217.

Appendix A: Artefacts produced during Chapter 5 Study

Appendix A1: Group A Interaction Design



'Car Theft'

Treatment

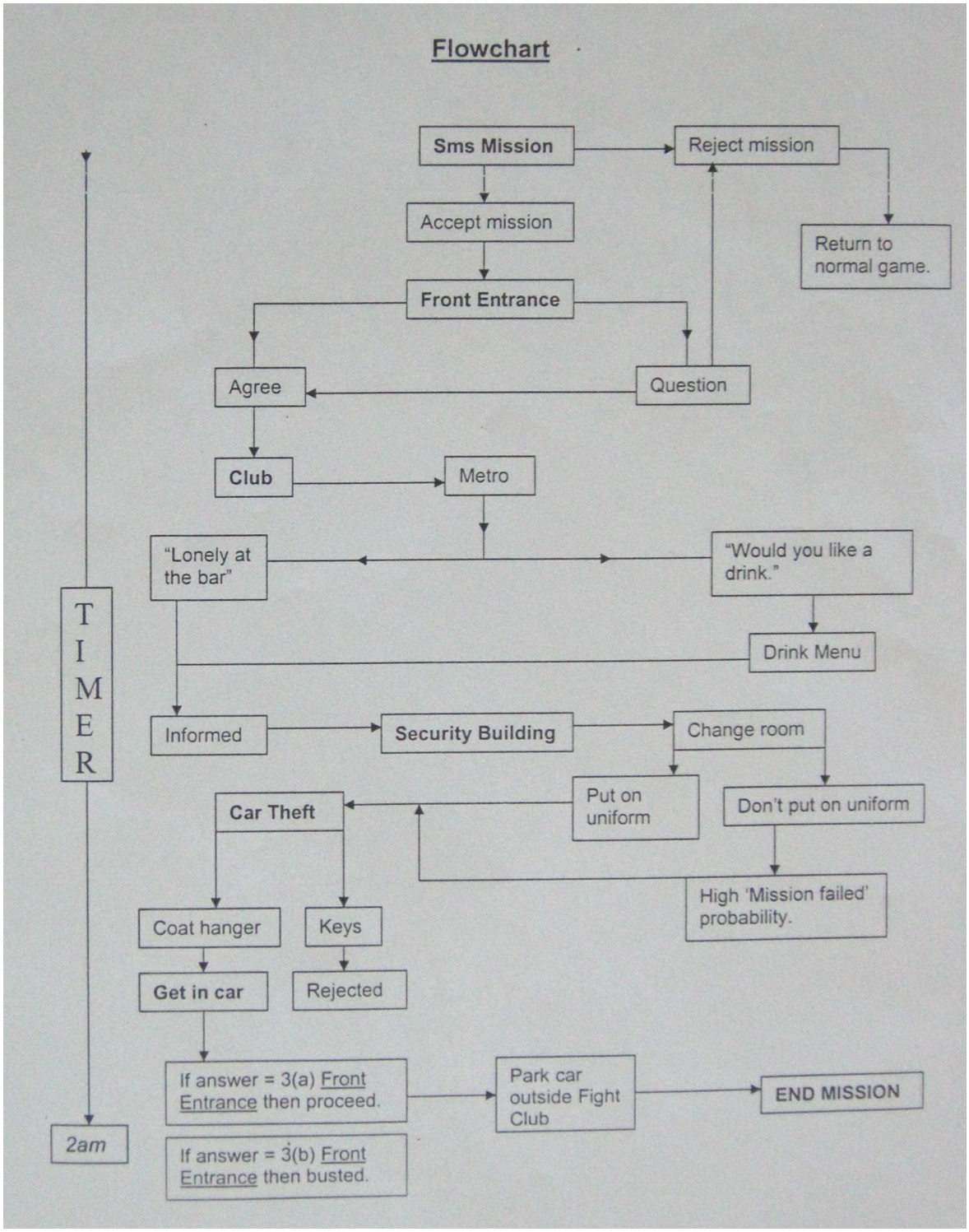
You have just won the main fight at the infamous Fight Club and are caught deep within the clutches of the underworld. Finally to prove yourself to the Under Lords you have to take on a mission. This mission entails dealing, sneaking, robbing and in order to seal the deal you are expected to steal a car with the goods needed by the Under Lords and to deliver it to them before 2am the next morning. The time is tight but with your thieving skills, you the player are able to beat the time and become the big shot of the under world.

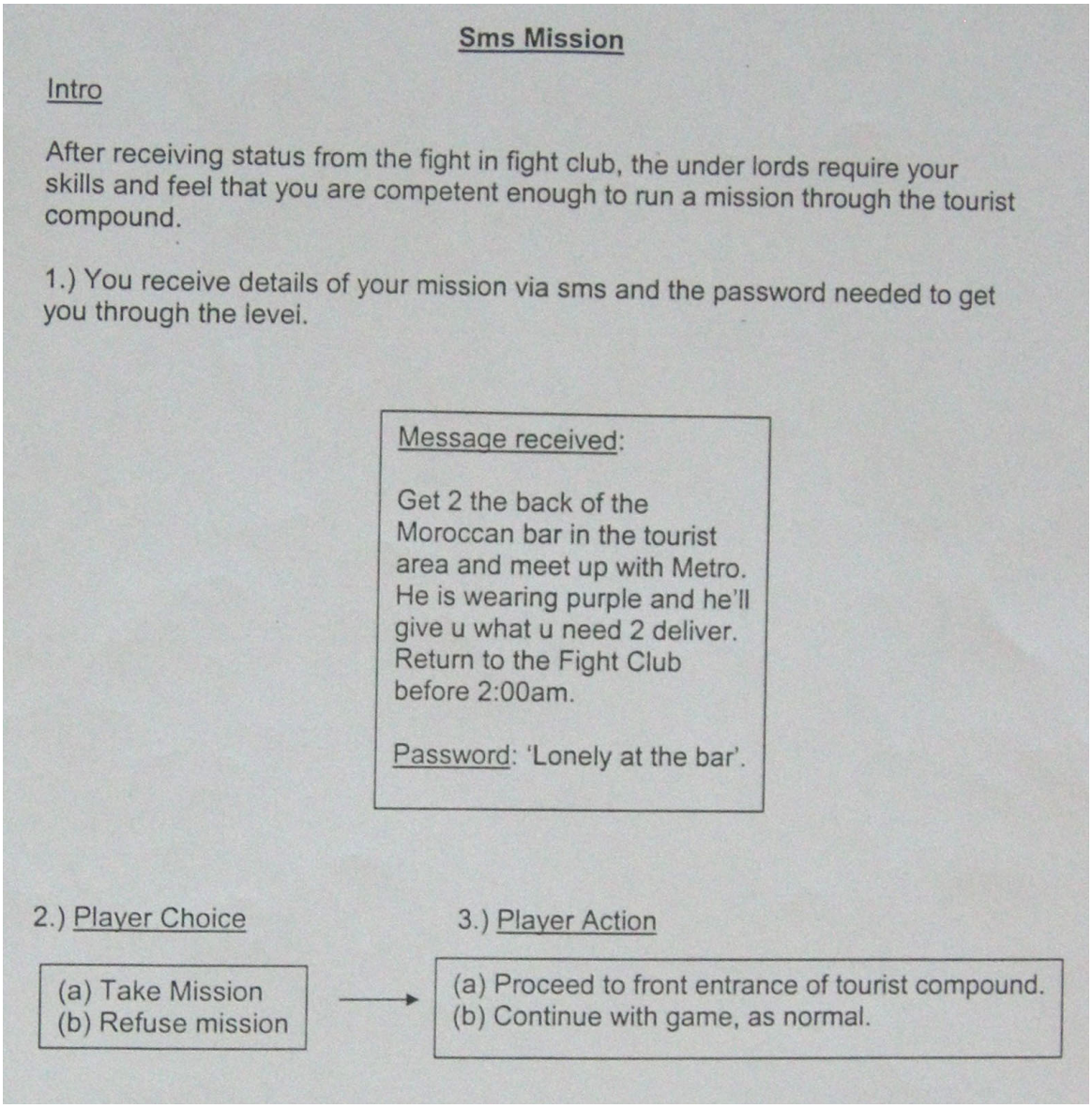
This mission is designed for a thief and your tourists have gone to bed or have dispersed into the nightlife of Handover City. Firstly receive a sms containing a briefing and a password for your mission. Take this information and proceed to the front entrance of the tourist complex. Gather information from the Security guard who seems reluctant to let you in and head to the Moroccan Club. In the club you will look for a fellow named Metro who is dressed in purple and comes across as very professional, at his job. After an exchange of the password, Metro reveals the location of the goods and depending on the way you treat him, he will give you a few tips on how to get out of the complex without being caught and will give you a security guard access card. The goods are placed in a red security car located outside the security building, where you the thief with the help of the access card have to steal an outfit, as a disguise to get into the car without being detected. You have an option of taking the car keys or a coat hanger once you have put on the security guard's outfit. These will determine if you get into the car or not. The thief then locates the vehicle and uses either the coat hanger to break in or the keys to open the door. They are the wrong keys and the coat hanger fits perfectly, but be careful that no one sees you using the coat hanger or you will be busted. Drive the car out of the complex and make sure that you deliver it to the Under Lords before 2am or the mission is failed.
Good luck!

Characters

Metro: He got his nickname from being the middle man in the deal and is the night train of the underworld, conducting the exchange of each deal and making sure they reach there destination. He is a short man, just able to rest his elbows on the counter, suave and clean cut. His attitude is set on his work and if you treat him well he is the type of guy that will lead you in the right direction and make you the player, radiate in the glassy eyes of the Under Lords.

Security Guard: He is tall and built mean and has dark features to match his 'hard ass' attitude. Unfortunately he is the stereotype of your underpaid uneducated bouncer with a short temper and an attitude that will take him as far as he has gone in life, searching people at a security gate. He commands respect as 'keeper of the gates'.


Flowchart
Sms Mission
Reject mission
Accept mission
Return to normal game.
Front Entrance
Agree
Question
Club
Metro
"Lonely at the bar"
"Would you like a drink."
Drink Menu
TIMER
Informed
Security Building
Change room
Put on uniform
Don't put on uniform
Car Theft
High 'Mission failed' probability.
Coat hanger
Keys
Get in car
Rejected
If answer = 3(a) Front Entrance then proceed.
Park car outside Fight Club
END MISSION
2am
If answer = 3(b) Front Entrance then busted.



Sms Mission

Intro

After receiving status from the fight in fight club, the under lords require your skills and feel that you are competent enough to run a mission through the tourist compound.

1.) You receive details of your mission via sms and the password needed to get you through the levei.

Message received:

Get 2 the back of the Moroccan bar in the tourist area and meet up with Metro. He is wearing purple and he'll give u what u need 2 deliver. Return to the Fight Club before 2:00am.

Password: 'Lonely at the bar'.

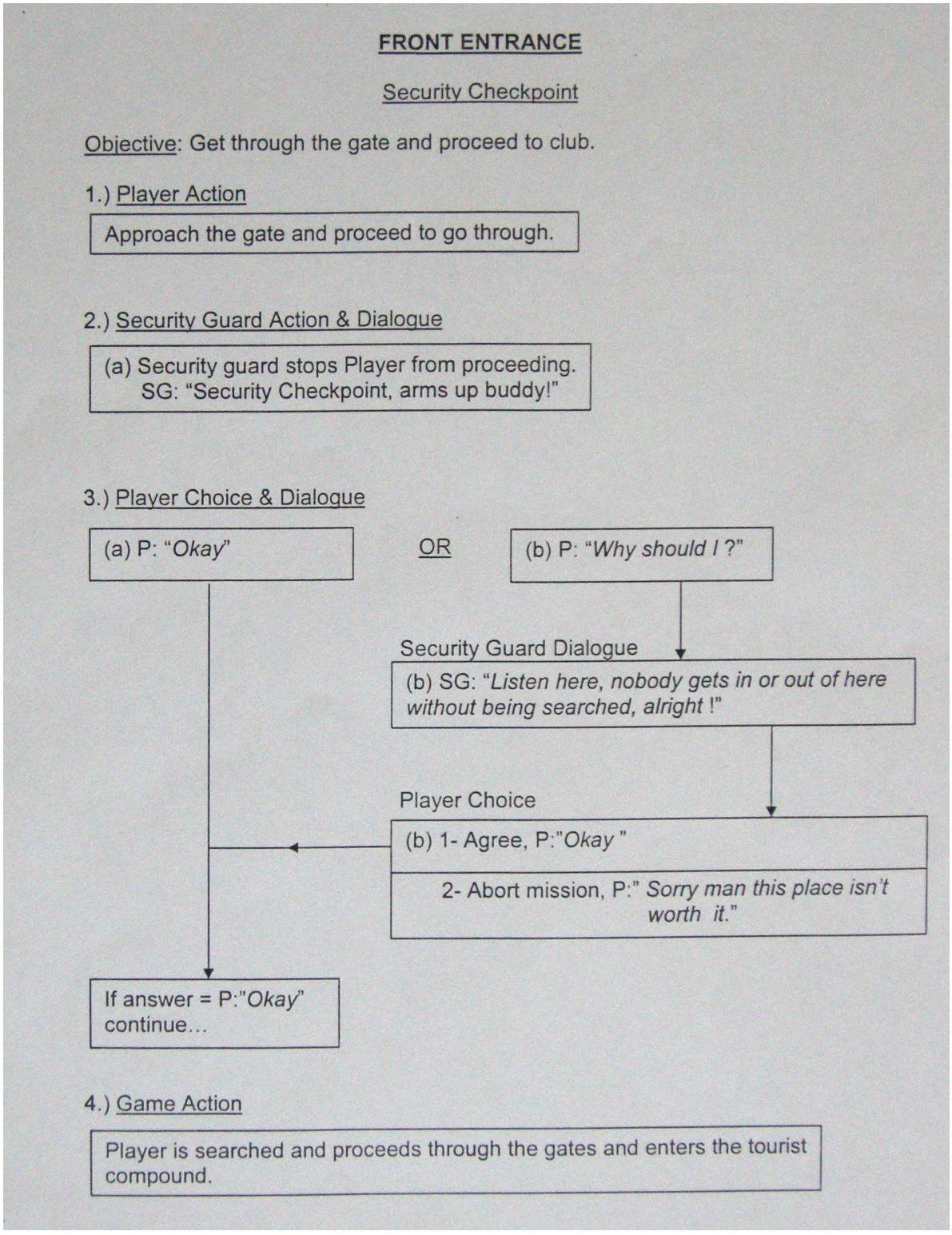
2.) Player Choice

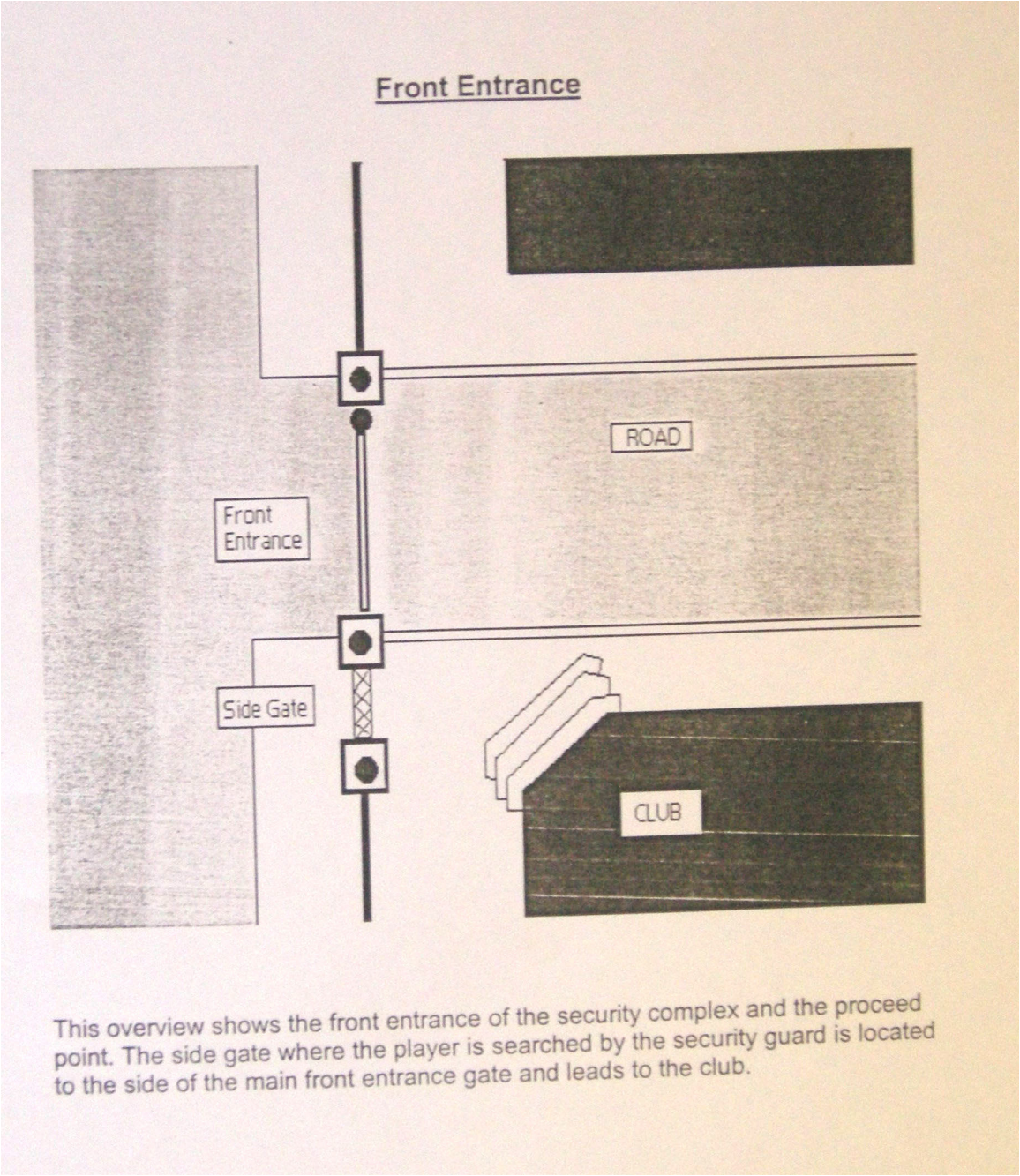
(a) Take Mission
(b) Refuse mission

→

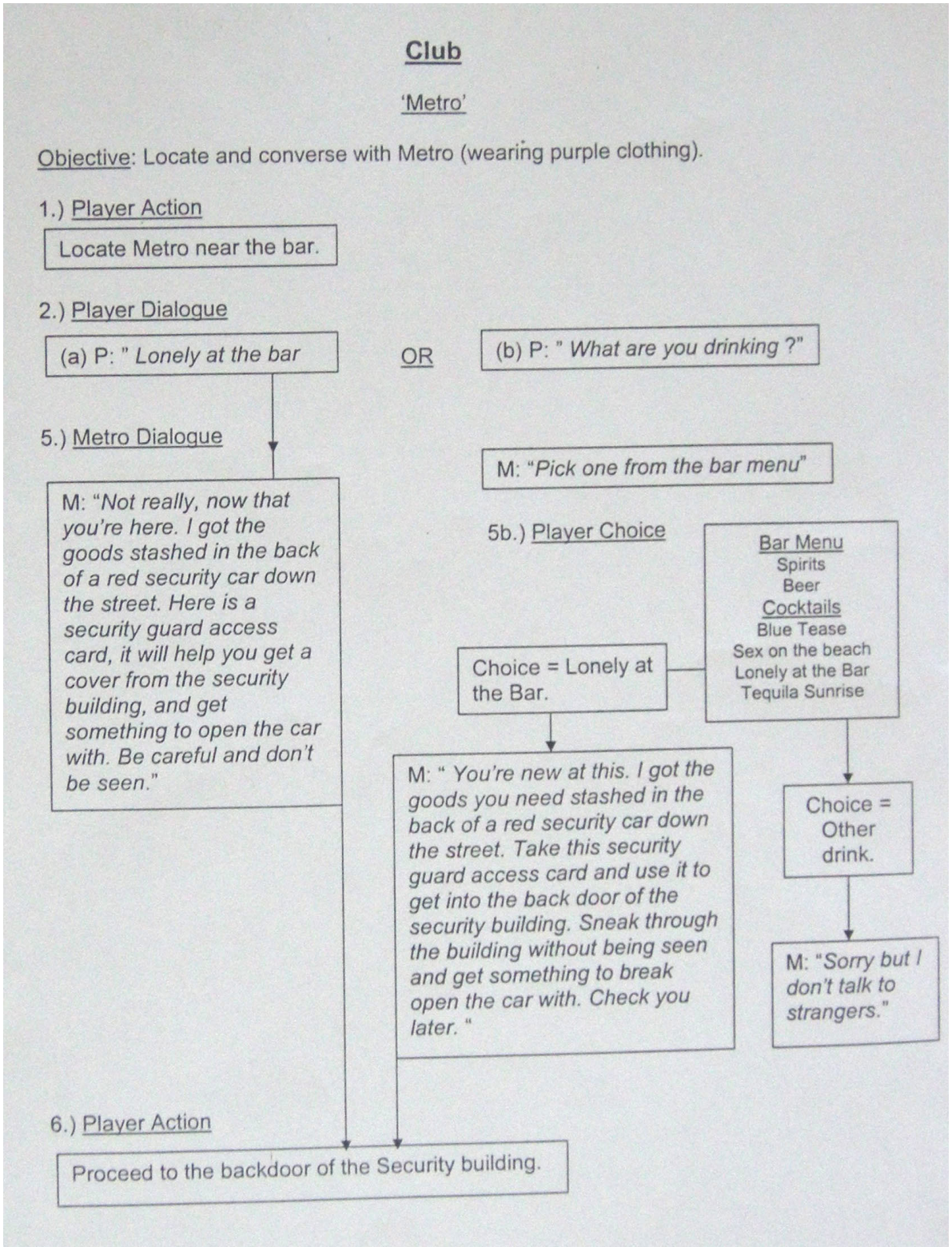
3.) Player Action

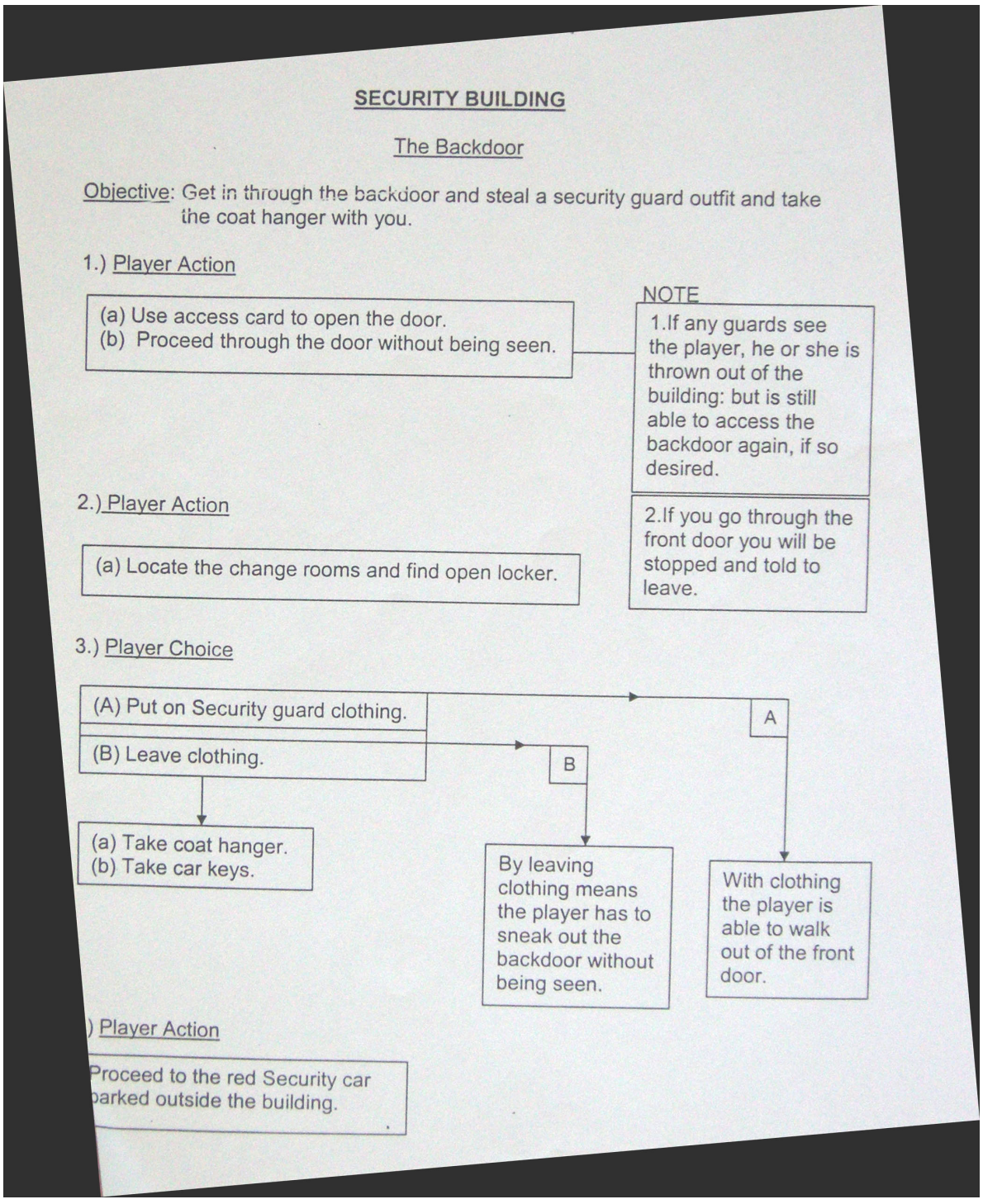
(a) Proceed to front entrance of tourist compound.
(b) Continue with game, as normal.


FRONT ENTRANCE
Security Checkpoint
Objective: Get through the gate and proceed to club.
1.) Player Action
Approach the gate and proceed to go through.
2.) Security Guard Action & Dialogue
(a) Security guard stops Player from proceeding.
SG: "Security Checkpoint, arms up buddy!"
3.) Player Choice & Dialogue
(a) P: "Okay"
OR
(b) P: "Why should I ?"
Security Guard Dialogue
(b) SG: "Listen here, nobody gets in or out of here without being searched, alright !"
Player Choice
(b) 1- Agree, P:"Okay "
2- Abort mission, P:" Sorry man this place isn't worth it."
If answer = P:"Okay" continue...
4.) Game Action
Player is searched and proceeds through the gates and enters the tourist compound.



This overview shows the front entrance of the security complex and the proceed point. The side gate where the player is searched by the security guard is located to the side of the main front entrance gate and leads to the club.


Club
'Metro'
Objective: Locate and converse with Metro (wearing purple clothing).
1.) Player Action
Locate Metro near the bar.
2.) Player Dialogue
(a) P: " Lonely at the bar
OR
(b) P: " What are you drinking ?"
5.) Metro Dialogue
M: "Not really, now that you're here. I got the goods stashed in the back of a red security car down the street. Here is a security guard access card, it will help you get a cover from the security building, and get something to open the car with. Be careful and don't be seen."
M: "Pick one from the bar menu"
5b.) Player Choice
Bar Menu
Spirits
Beer
Cocktails
Blue Tease
Sex on the beach
Lonely at the Bar
Tequila Sunrise
Choice = Lonely at the Bar.
M: " You're new at this. I got the goods you need stashed in the back of a red security car down the street. Take this security guard access card and use it to get into the back door of the security building. Sneak through the building without being seen and get something to break open the car with. Check you later. "
Choice = Other drink.
M: "Sorry but I don't talk to strangers."
6.) Player Action
Proceed to the backdoor of the Security building.



SECURITY BUILDING

The Backdoor

Objective: Get in through the backdoor and steal a security guard outfit and take the coat hanger with you.

1.) Player Action

(a) Use access card to open the door.
(b) Proceed through the door without being seen.

NOTE

1.If any guards see the player, he or she is thrown out of the building: but is still able to access the backdoor again, if so desired.

2.) Player Action

(a) Locate the change rooms and find open locker.

2.If you go through the front door you will be stopped and told to leave.

3.) Player Choice

(A) Put on Security guard clothing.

(B) Leave clothing.

A

B

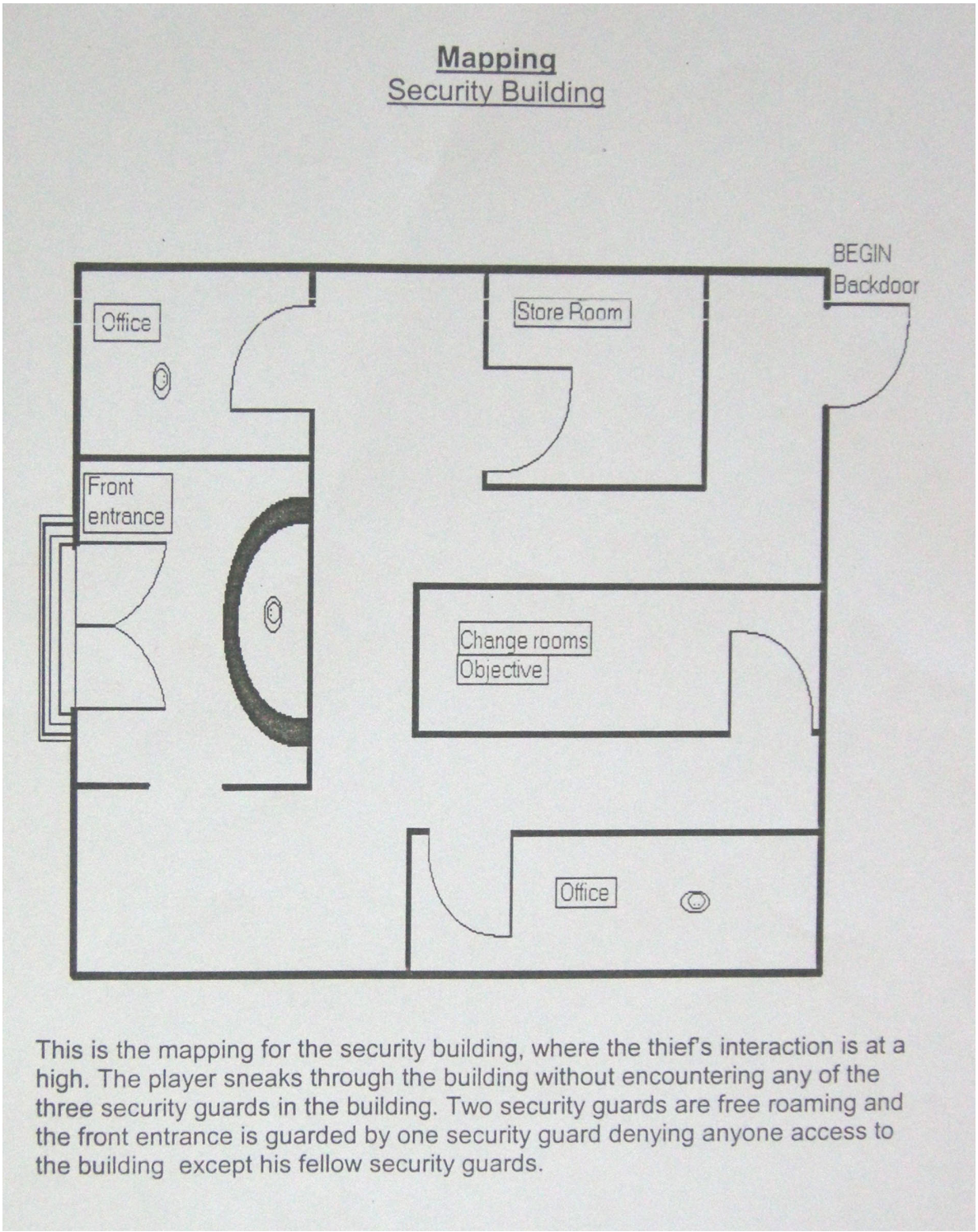
(a) Take coat hanger.
(b) Take car keys.

By leaving clothing means the player has to sneak out the backdoor without being seen.

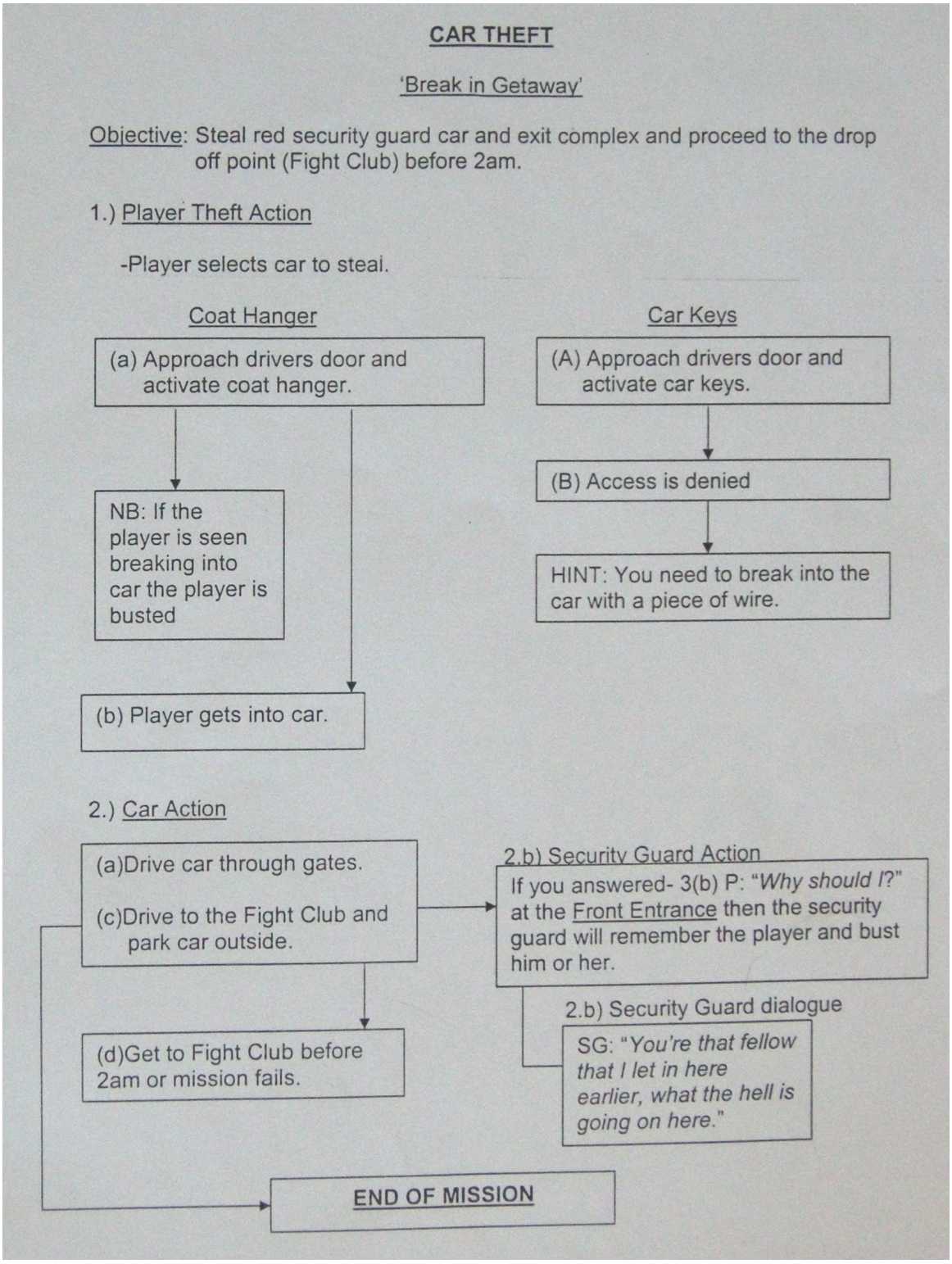
With clothing the player is able to walk out of the front door.

) Player Action

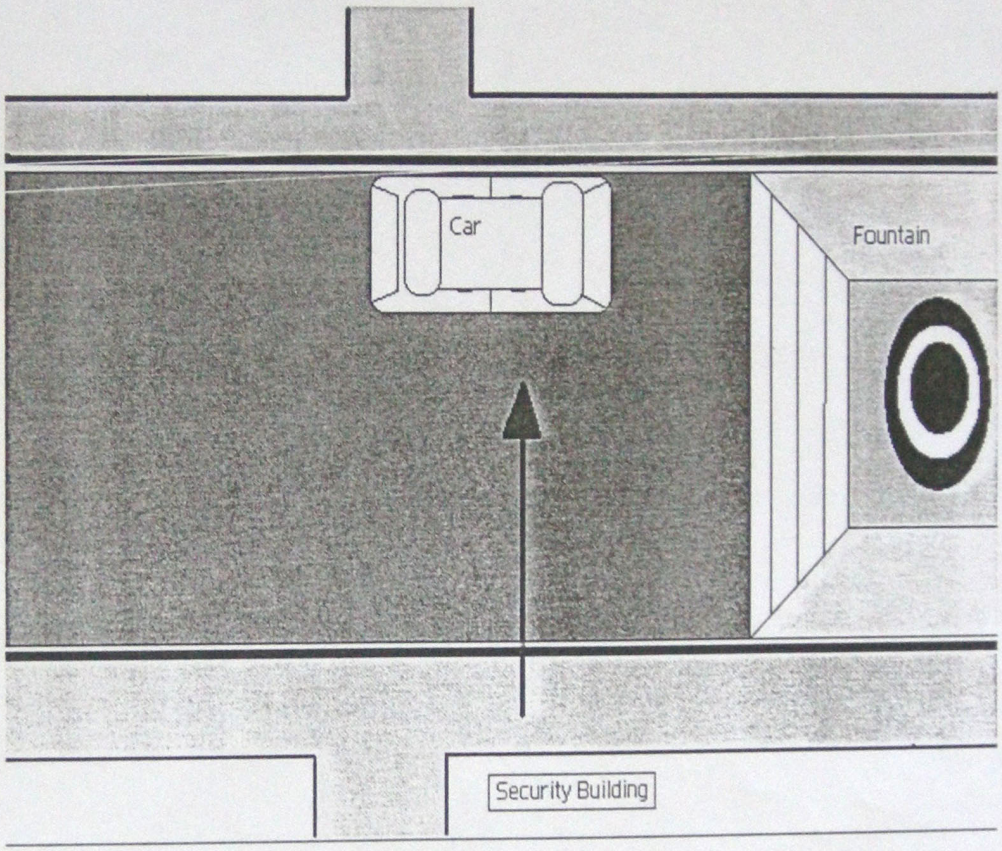
Proceed to the red Security car parked outside the building.



This is the mapping for the security building, where the thief's interaction is at a high. The player sneaks through the building without encountering any of the three security guards in the building. Two security guards are free roaming and the front entrance is guarded by one security guard denying anyone access to the building except his fellow security guards.


CAR THEFT
'Break in Getaway'
Objective: Steal red security guard car and exit complex and proceed to the drop off point (Fight Club) before 2am.
1.) Player Theft Action
-Player selects car to steal.
Coat Hanger
(a) Approach drivers door and activate coat hanger.
NB: If the player is seen breaking into car the player is busted
(b) Player gets into car.
Car Keys
(A) Approach drivers door and activate car keys.
(B) Access is denied
HINT: You need to break into the car with a piece of wire.
2.) Car Action
(a)Drive car through gates.
(c)Drive to the Fight Club and park car outside.
(d)Get to Fight Club before 2am or mission fails.
2.b) Security Guard Action
If you answered- 3(b) P: "Why should I?" at the Front Entrance then the security guard will remember the player and bust him or her.
2.b) Security Guard dialogue
SG: "You're that fellow that I let in here earlier, what the hell is going on here."
END OF MISSION

Car Theft



The car theft continues from the security building task and the red arrow shows the root taken by the player to gain access to the car. The player must stand on the drivers side of the vehicle

Sound

The music down tempo, with an atmospheric mix of strings to create a sense of the mood combined with a hard, sampled drum loop (a la Portishead) and James Bond style horns and bass-line. The tempo picks up as the player starts to run out of time on final hi-jacking task.

Appendix A2: Group B Interaction Design

Welcome to Club Dune

Gameplay

Player's will be able to select attributes that will affect how the game itself is played and the appearance of the central character. The 'thug' is brash and solves problems via brute force. Consequently, his strength and fighting ability are superior to the 'sneak' and 'con'. The 'sneak' has stealth and problem solving on his side. He is far more adept at utilising certain gadgets and attacks with speed and precision. The 'con' does not gravitate towards confrontation. The player will be required to don disguises and quite literally 'con' opponents with the aid of trickery and deceit. The confrontations in the game are somewhat different, rather than simply obliterate and opponent with punches, kicks and weaponry; players will be required to use alternate means of combat e.g. using mirrors to find the position of targets. This will aid in ensuring that players will have 'something new' to look forward to every time the game is played.

Player Experience

The player has been given the task of retrieving a package from Club Dune for gang lord and business man Mr V. The package contains the highly sort after MA-GK, a sophisticated weapon which has the ability to fire multiple forms of ammunition and accommodate different calibre ammunition. The player will have to find his way into the club by choosing out of two possibilities. The main entrance or the Staff entrance. The entrance which the player chooses provide the different challenges will have different consequences. The player will have text based interaction with the different characters in the environment. Some will help you achieve your goal while others are prepared to kill you.

Upon entering the club the player will need to find his helper who comes in the form of Natasha. Natasha will provide the player with information essential in completing the mission and staying alive. The player will interact with a few characters inside the club which is designed to give the player the night club experience which Cape Town is famous for. Unfortunately your task in the club will not be trying to get lucky with an attractive female or dance to the music. The player will make their way to the VIP room of the club to steal the package. Because this is no ordinary package the player will have to fend off a number of challenges from some armed gangsters. These gangsters have come to club to make a bid for the package you're stealing in an auction

being held at the club. The sleek and funky atmosphere of the club will then be transformed into a battlefield as each person earns the right to have the package by fighting for it. The player will engage in a gun battle against Cape Town's most hardened gangsters and try to make it out alive and get paid. The player will find the game switching from a stealth and witty scenario to a violent action packed gun battle where it's either them or you that wins or dies. This stage will provide the player with the chance to have a full club experience buy allowing a wide range of interaction and challenging objectives that will suit almost all types of game players.

Narrative, Plot and Interactions

Mr V has been monitoring the activities of 'Gang X', the gang that has taken over the running of Club Dune. Club Dune is a sophisticated but highly dangerous club that attracts Cape Town's party goers and beautiful people, but also caters for the needs of Cape Town's hardened gangsters, whether it be weaponry or drugs. Tonight however is a special night and no one knows this more than Mr V. Tonight the Club will be hosting an auction. The auction will be for a new weapon that has been developed by the Russian Military. The weapon is called the MA-GK which is short for "Magic Stick". The weapon got its name from the unique ability to fire different calibre ammunition rounds. It is the only handgun able to accommodate .38, .45 calibre and 357 ammunition. The weapon is also equipped with cylinders which can fire armour piercing and explosive rounds. These features make this weapon a very valuable commodity for Cape Town's gangsters wanting to get ahead of the competition and the Police.

Mr V has other plans for this weapon as he knows that not only will the gun be on auction but also the design documents of the weapon meaning, whoever has the means of producing the weapon can mass produce this awesome weapon and supply whoever he likes and at a high price. Mr V is trying hard to beef up his image as an influential business man and like a good business man he has seen a very 'legitimate' and lucrative opportunity. He will produce and supply this weapon to South African National Defence Force. He has held secret meetings with shadow agents of the Defence Force who have guaranteed him the contract to produce the weapons if he is able to get the design documents. The contract is worth R300 million. The Defence Force will save millions of Rands by buying the weapon locally rather than paying the Russians for the weapon. If Mr V can produce the goods, they'll keep quiet and pay up. With the state of the art weapon in his possession Mr V will become the most powerful man in Handover. Mr V cannot be shown to have anything to do with the stealing of the weapon and that is why you the user have been called in. You have been seen to be the ideal candidate for the mission because one of Mr V's henchmen, Langer, who saw your need for some money after he watched you mug a lady for her necklace. You then approached Langer to try sell the necklace to him but Langer's long experience in the

gang world told him that you are the right man for the job. Langer gives you a cell phone and tells you to be at Club Dune tonight at 23:30. You will receive instructions from Langer once you start the mission outside the club.

Mr V has been monitoring the club for some time and he has done this by having one of his agents infiltrate the club by going undercover and securing a position as the club PR and Hostess. Her name is Natasha and she has managed to get information on the packages location and when the ideal opportunity to steal the package. The need to not connect this mission to Mr V means that Natasha cannot be seen to have any link to the mission. The user has been hired to do the job so as to not incriminate Natasha and Mr V, they obviously don't care if you live or die. It's all about the package. The user will begin the standing outside the main entrance of the club. The user will hear the beep of his cell phone and a Cell phone screen will appear. The message will cover the bottom half of the screen and will be a message from Langer.

"Find a way into the club, and once your inside you must find Natasha. She will provide you with information on the package and how to get it."

The player will press a button to close the SMS screen and is ready to begin the mission.

It looks like an ordinary night at the hip and funky Club Dune, the bouncer is at the main door and in the cue will be two women about to go in the club. If you have money you can simply pay the bouncer, but if you do not you'll have to find a way inside. You could try talk to the two ladies standing in the cue. Bouncers are vulnerable to the pleas of beautiful women and joining these women will win you the bonus of getting into the club for free. Your success will depend on your player attributes. There is different interaction for whichever attributes dominate your player. The Thug, Con or Sneak.

Interaction

Main Entrance:

When the player moves within 1m of the women "Press Button" will appear on the screen. When the player presses the button the shot will CUT to a CLOSE-UP SHOT from Players POV of the two women. The textual interaction will begin and the player will select with the arrow keys.

Option: Thug

Women: "Hey big boy?"

Player will receive a number of options to reply with.

Option 1: "You girls look like you could use some company."

Option 2: "Want to feel my muscles?"

If Player chooses Option 1.

Women: "Wow, I didn't think we looked that desperate."

Interactions end. Player will move back 1 m from women and will have to try again or find an alternative method.

If Player chooses Option 2.

Women: "Don't embarrass yourself sweetie."

Option: Thug

Option 1: "Come on, I'll show you mine if you show me yours."

Option 2: "Whatever, I've got bigger fish to fry."

If Player chooses Option1.

Women: "Hmm... Lets see how its goes."

Player succeeds in gaining entrance to the as there is a cut scene. MEDIUM SHOT of Player walking into Club Entrance.

If Player chooses Option 2 the interaction will end and the player will resume 1m away from the women.

Option: con man

Women: "Well hi there handsome..."

Option 1: "You ladies are looking gorgeous tonight; care to come inside for a drink?"

Option 2: "If you Ladies would like to come inside with me, it would make things much easier."

If Player chooses Option 1:

Women: "Sure, why not.

Cut to Medium Shot of Player entering the club with women.

If player chooses Option 2:

Women: "Alright then handsome, we'll get you in."

Cut Medium Shot of player entering the club with the women.

Option: Sneak

Women: "Can we help you?"

Option1: "Hi I-I was wondering if you'd like to come inside with me?"

Option2: "Excuse but I think that guy is calling you."

If Option 1: "Maybe some other time." Player resumes 1 m away from women.
If Option 2 there will be a CUT to CLOSE UP Shot of women looking away and then CLOSE UP of Player's hand steals a Comp from woman's bag.
CUT to Medium SHOT of player walking into club.

The player can try to approach the bouncer and try to get in by interacting with him.
Once the player come within 1 m of bouncer, there will be a cut to a CLOSE UP of the Bouncer and text will appear.

Bouncer: "Sorry but there is a private party tonight."

The Bouncer will not budge unless you have more than R200 on you. If the player has more than R200 he will be able to enter.

Player: "I'm sure you can make an exception."

CUT to CLOSE UP shot of player's hand slyly giving the bouncer the money. Then Medium Shot of player walking into the club.

Staff Entrance

Should the player look for another way of getting inside the club the other option will be the STAFF Entrance. This will be guarded by the twin brother of the bouncer at the main entrance. Outside the Staff entrance there will be a staff member taking a cigarette break near the trash bins. He'll have a brightly coloured Staff pass attached to his wrist. You can mug the worker by approaching him and pressing a button which will make the player knock the worker unconscious and hide the worker behind the trash bins. The player will then be wearing the Staff pass on his wrist and will be able to enter the club via the Staff entrance. When you approach the door there will a CUT to a MEDIUM SHOT of the player walking through the door with 'STAFF' shining above the door.
If the player approaches the Staff Entrance without having the Staff pass the bouncer will turn the player away by saying, "Sorry, only Staff get in here." The camera will move to a Front on Medium shot of the bouncer as he turns the player away. Using this entrance will have consequences. The worker you knocked out will wake up after a certain period according to the attributes of your player. When the worker awakes he will warn the bouncer and this will result in him coming inside

the club to look for you. This will be shown by a Medium Shot of the Bouncer opening the Staff door and walking inside the club.

Thug *The thug will pack a stronger punch that the Con or Sneak and so the worker will be knocked for 3 minutes until he warns the bouncer and he comes looking for. Once the bouncer is inside the club the only way he can catch you is if you are within 3m and in front of him. Once the player has stolen the leatherjacket and sunglasses needed to get into the VIP section the bouncer will not be able to recognise you and so the threat will cease to exist. If the bouncer catches you before you can get the jacket and glasses the game will be over and the player will restart outside the club.*

Con *The Con will knock the worker unconscious for 2 minutes. The cut scene showing the bouncer*

entering the club will be the same for all attributes, but the Con will only have to be within 2m of the bouncer to be spotted and the same rule applies when the player has stolen the jacket and glasses.

Sneak *The sneak will only have 1 minute before the bouncer enters and poses a threat. The upside for the sneak is that he will only be seen if he is 1 m in front of the bouncer and so will be able to move more freely than the other attributes. Like the Con and Thug, the bouncer will not see him once he has the jacket and glasses.*

Your first objective is now complete and you have entered the club. You will start at the main entrance regardless of the entrance you decide to use. Next to you will be the dance floor and the bar will be next to it a few metres away. The player will need to find Natasha to get the information on how to go about getting the package. As in most places such as Club Dune the best place for information on where to find somebody is the bar. The user can choose to walk around the club to try find Natasha but the quickest solution will be to walk to the bar. Once the player is within 1 m of the bar interaction can begin by pressing a button. Interaction will be the same regardless of the player attributes.

CUT to Close Up of Barmen, Player POV.

Barman: "What can I get you?"

CUT to Close Up of Player.

Option 1: "I'm looking for Natasha."

Option2: "Could you show me the bathroom."

Option 3: "I'll have a Vodka Tonic."

If Option 1 CUT to Close Up of Barmen.

Barman: "There are a lot of Natasha's here man."

Option 1: "The one I'm looking for is the hostess."

Option 2: "This R20 tip should clear your memory."

Barman if Option 1:"I don't know what you're talking about."

End of interaction, player resumes 1 m away from the bar.

Barman if Option 2:"She's the lady holding the clipboard near the bathroom."

PAN right to Medium shot of Natasha. End of interaction player resumes 1 m from bar.

If Option 2 Camera PANS Right to Long Shot of Toilet sign. Natasha is standing under it.

Barman: "It's over there, next to where the hostess is standing."

End of interaction. Player resumes 1 m away from Bar.

If Option 3 CUT to Close Up of player.

Option 1: "Pay the barman and leave."

End Interaction.

Option 2: "Hey, I'm looking for Natasha."

Barman: "She's over by the bathroom holding the clipboard."

PAN Right to Medium Shot of Natasha. End of interaction.

With Natasha located its time to get instructions on how to retrieve the package. The player will walk towards Natasha. She will see and when you are within 2m of Natasha she will walk towards the back of the club and then there will be a CUT Scene. High Angle Medium Shot of Natasha walking through a doorway and the sign above the door will read "Storage." The player must then make his way to that store room to talk to Natasha. When the player reaches the door he will press a button to enter the room.

CUT Scene to MEDIUM REAR SHOT of Natasha inside the room and Player walking in through doorway. Cut to Low Angle Shot of Natasha.

Natasha: "Hi I'm Natasha, you don't have much time, and you'll first have to change those clothes if you don't want raise suspicion. Steal a leather jacket and sunglasses, that's how the gang members like to dress. Meet me back here when you've changed, I'll have everything ready."

Cut to Medium Shot of Player walking out of room. Player continues outside the store room. Player must look around the club for the leather jacket and sunglasses. Items will be in the far corner of the club. The will be shown by two spotlights that will be shining over them. The player will approach the items and will steal them by pressing a button. The bouncer from the Staff entrance will know be alert and will be patrolling around the club looking for you. You'll have to avoid him and steal the jacket and glasses. Once you have stolen the items you will need to go back to the store room and consult with Natasha.

When you approach the store room door the same Cut scene will show once you enter but the player will know be wearing the leather jacket and glasses. Cut to a High Angle Medium Shot of Natasha. Natasha will have different instructions for the Sneak character.

Thug and Con.

> *Natasha: "Good work, the VIP section is upstairs, when you get up there tell the bouncer that, "You're here to see the Magic Show." He'll let you through, once you are in the lounge look for the head office, the package is in the safe. The combination is 15 - 34 - 50. I will distract the club owner while you get in and get the package. You have very long so open the safe and get the package. Some gang representatives will be arriving soon, so be ready. When you have the packages meet me outside the Staff entrance."*

Option: Sneak

> *Natasha: "Good work, the package is inside the VIP section, in a safe in the head office. You can climb up this ventilation duct.*

(Camera moves PANS Downward from High Angle Shot to LOW Angle, revealing the Ventilation duct.)

> *You can by pass the bouncer and land in the office. The combination for the Safe is 15 - 34 - 50. I'll distract the club owner long enough for you to get in and get the package. Once you have it come back down and meet me outside the Staff entrance."*

VIP LOUNGE:

Options: Thug and Con

The Thug and Con Artist will resume outside the store room and will have to proceed up the stairs and the entrance to the VIP lounge will be a Red Curtain with a large mean looking bouncer called

'Stop Sign'. The player will approach the bouncer and once he selects to interact with the bouncer there will be CUT to LOW ANGLE CLOSE UP of Stop Sign from the player's POV.

Stop Sign: "What can I help you with?"

Player: "I'm here for the magic show."

Zoom out to Medium Shot of Stop Sign opening the curtain.

Stop Sign: "It begins at midnight, but you can wait in here."

Cut to Medium Shot of player walking inside VIP lounge.

If the player attempts to enter the VIP lounge before seeing Natasha and collecting Jacket and Glasses, Stop Sign will respond differently.

Stop Sign: "The party is downstairs!"

If player has collected the jacket and glasses but has not returned to the store room to get the password and the combination for the safe the player will have different options to respond with.

Stop Sign: "What can I help you with?"

Option 1: "I'm looking for Natasha."

Option 2: "Is this the VIP lounge?"

If player chooses Option 1

Stop sign: "Natasha is not up here."

End of Interaction; resume 2m away from the bouncer.

If player chooses Option 2

Stop Sign: "The party is downstairs!"

End of Interaction; resume 2m away from the bouncer.

Once the player enters the VIP lounge there will be a CUT scene. Players POV of Natasha walking out of the head office followed by the club owner. The player must then proceed into the head office.

Options: Sneak

The Sneak will resume inside the store where the grill blocking the ventilation duct has been opened. The user will move to the grill, by pressing the 'use key' they will grab onto the grill and the 'up' will make the player climb into the ventilation duct. There will be a cut and the player will be inside the ventilation duct. The view will be a first person view from the players perspective, as the player moves along the tunnel will lighten and the player must make his way to the head office. When the player reaches the office he will approach the ventilation duct. CUT to High Angle Long Shot of the head office. Club owner is sitting at his desk and the Safe is in the corner of the room. Natasha walks into the office and the owner gets up from his desk and follows Natasha out the office. Cut to Low Angle Medium long Shot of the ventilation duct, user opens the duct and gets down. Player resumes the game inside the head office.

Head Office:

Once the player is in the office a clock will come up at the top of the screen showing that the player has one minute to open the safe. The player will walk towards the safe and will press the use button to begin opening the safe. The view will change to a Close Up of the Safe door and the dialling knob. A small screen showing the number that has been dialled will be above the knob. The player can change the number of the dial by using the arrows keys. There will be three lights above the number screen. Once the player gets the first number one of the lights will go green and then with the second number. All three numbers have been put in all three lights will be green and the safe door will open revealing the package and a 9.mm pistol.
There will be variations in the speed that the dial moves when the player is opening the safe. The Thug's dial will move faster than the other characters and will be harder to focus on the number. The Con's will move at a medium speed while the sneaks will move the slowest and should be easiest to open.

Once the safe is open the player will resume but this time he will have a backpack carrying the package and his weapon in his hand. Player will then walk out of the office and into the lounge. When the player is 2m from the office Stop sign will come into the office carrying a 9.mm pistol and he will begin firing at you. The player will press the aim and fire button to shoot. Player attributes will determine how many times the player will have to hit Stop sign to kill. Thug = 3, Con = 4 and Sneak = 5. Once the player kills stop sign he will move out of the VIP lounge. When the player gets to the exit of the lounge there will be a Cut scene. Medium Shot of Khan walking into the Club Entrance while other people are running out of the Club. Khan walks in with sinister music playing. He is also holding an A-k 47.

Cut to player who will resume at the top of the stairway. Player will move down to the bottom of the stairs which will trigger the Club owner who is downstairs to begin shooting at you. He also carrying a 9.mm pistol and the same rules apply to the club owner in terms of the number of times he must be shot before he dies. With the club owner out of the way the player must now kill Khan to have a clear passage to the Staff entrance. Khan will open fire with his A-k and spray bullets everywhere. The player will die and have to restart in the head office at the safe if he is shot more than six times as the power bar will indicate. Health will be in the form of alcohol bottles throughout the club. The bottles will be marked with a red cross and the player will just have to make it to one before Khan Blows him away. Unlike the club owner it will take more shots to kill Khan. Thug = 5 Con = 6 Sneak =7. The battle will be on the main floor of the club. Khan will move around the entrance and dance floor, while the player will be near the stairs and the rear of the club.

Once Khan has been dispatched the Club will be completely empty and the user will make his way to the Staff Entrance which will be a door with Staff Exit marked above it.
The user will press the use button when he is at the door and then Cut to Medium shot of player walking out the Club to find Natasha waiting in a car. Player climbs into the car which drives off. Congratulations. Mission Complete.

Characters

KHAN

The man known to the underworld as Khan is a notorious gunman who enjoys his work a little too much. His stylish, cool exterior hides a psychotic, drug-addicted mind. Khan is rumored to be doing the dirty work for a high profile politician, "Puma". He is totally unapproachable, excerpt when someone has something he wants e.g. a special blend of rum and coke/ a truckload of narcotics. Khan never leaves home without his custom made Kalashnikov .47. Khan is present at the club to bid for "Magic - Stick", a custom-built handgun that can accommodate nearly any caliber bullet. Khan will be the final obstacle as the player completes the scene.

NATASHA

Natasha is a powerful force in Mr. V's empire. She is almost unknown except in Mr. V's closest circles. She applied for the job at Club Dune under the alias "Natasha Davies". Her beauty and charm have helped her in gaining the trust of the ruthless gangsters who run Club Dune and they have shared the information about the auction. Natasha will be the players guide to completing the mission. She will provide the player with the codes to break into the Safe and the protocol needed

to enter the VIP lounge. Natasha will also be the player's getaway driver as they speed away from the club on having completed the mission.

Stop Sign

Stop Sign is picture of brute force and destruction. He has cemented his reputation and nick name Stop Sign by never letting an offender getaway and stopping anyone who tries to rip the club off. He stops them dead. Stop Sign will be guarding the VIP lounge and will be the first to pull his gun on you as you try to make your escape.

The Women

The woman you meet at the beginning of the game will be regular members of Club Dune. Their familiarity with the bouncers will mean that if you can impress them with the right words, they will get you inside the club.

Club owner

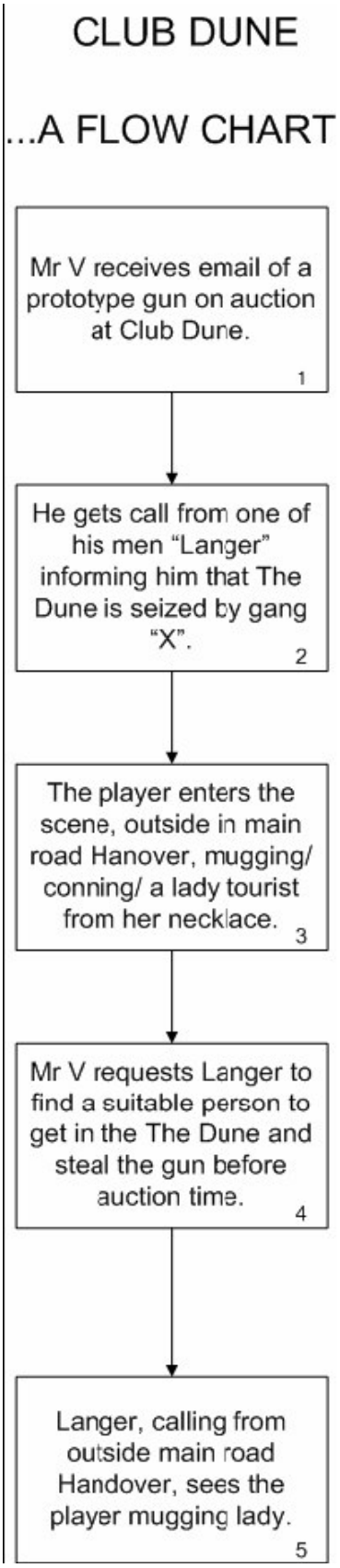
The gangster running the Club is Maxwell. He has taken a liking to Natasha and responds to her every call. She will distract the fickle owner and lead him away from the office while you steal the package. The player will have to kill Maxwell before completing the game.

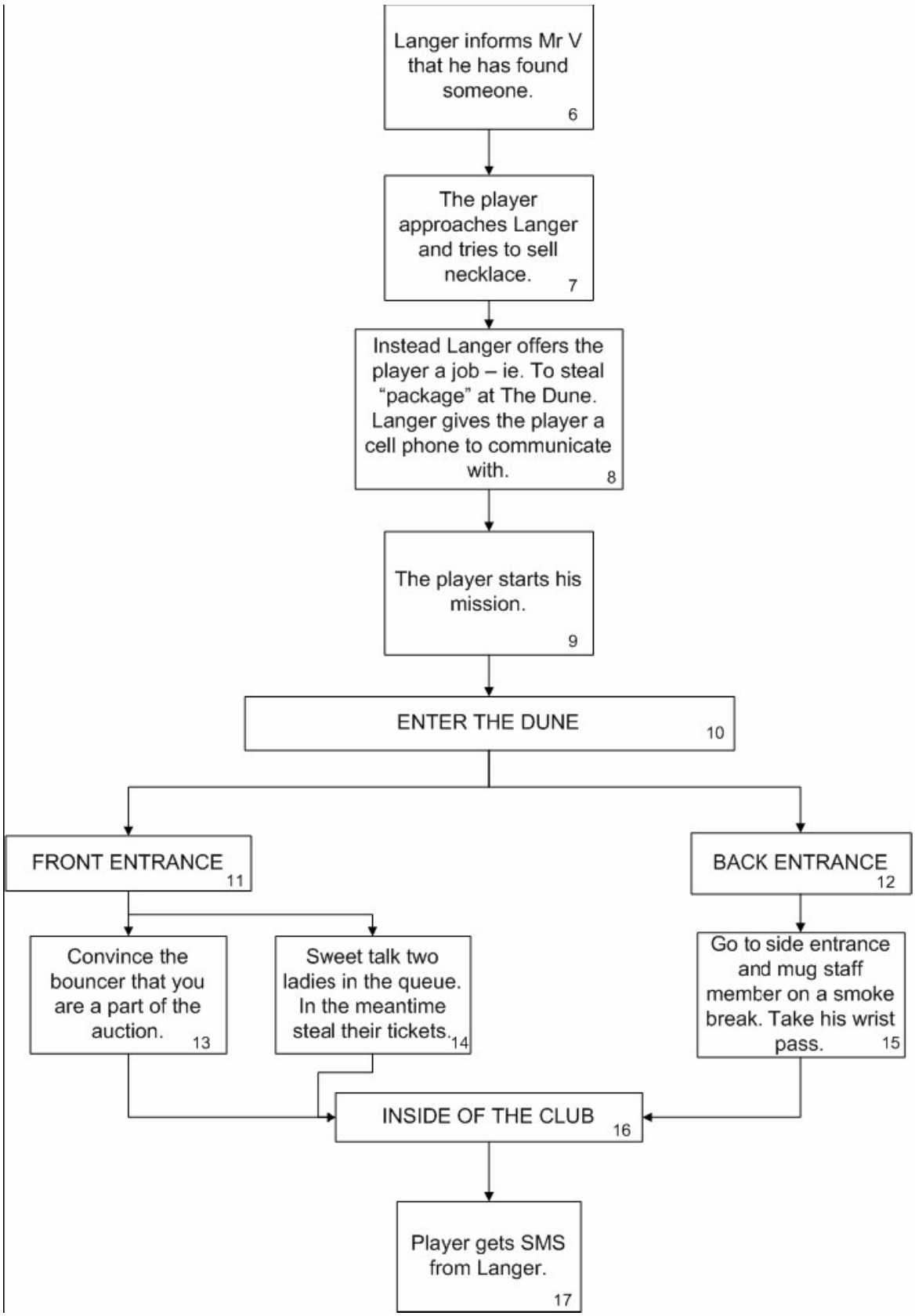
Barman

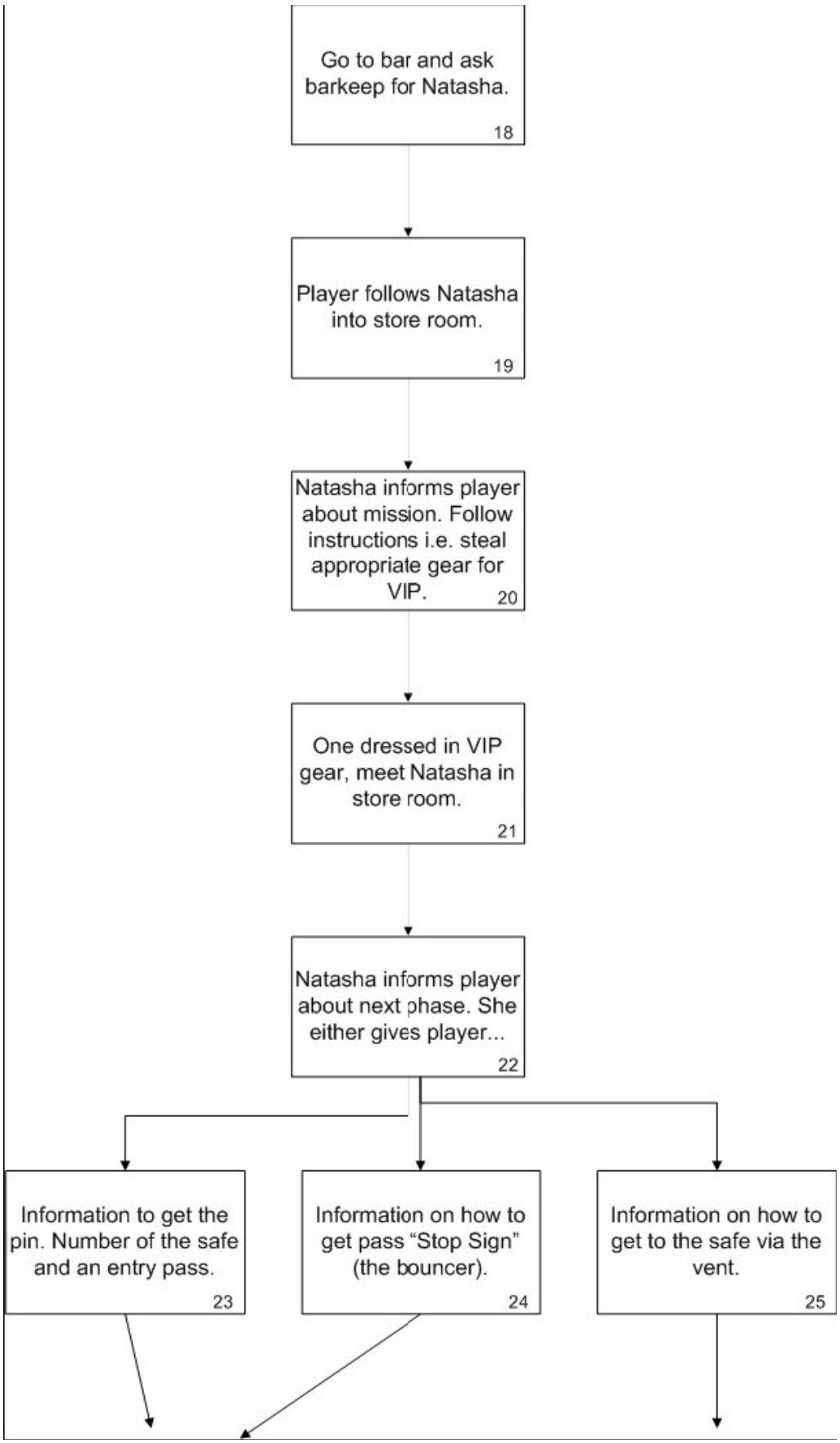
The bartender is your usual bartender who after working in the club every night has come to know the different people in the club. He knows who the Natasha you're looking for is, but you'll have to be a bit generous to get any information from him.

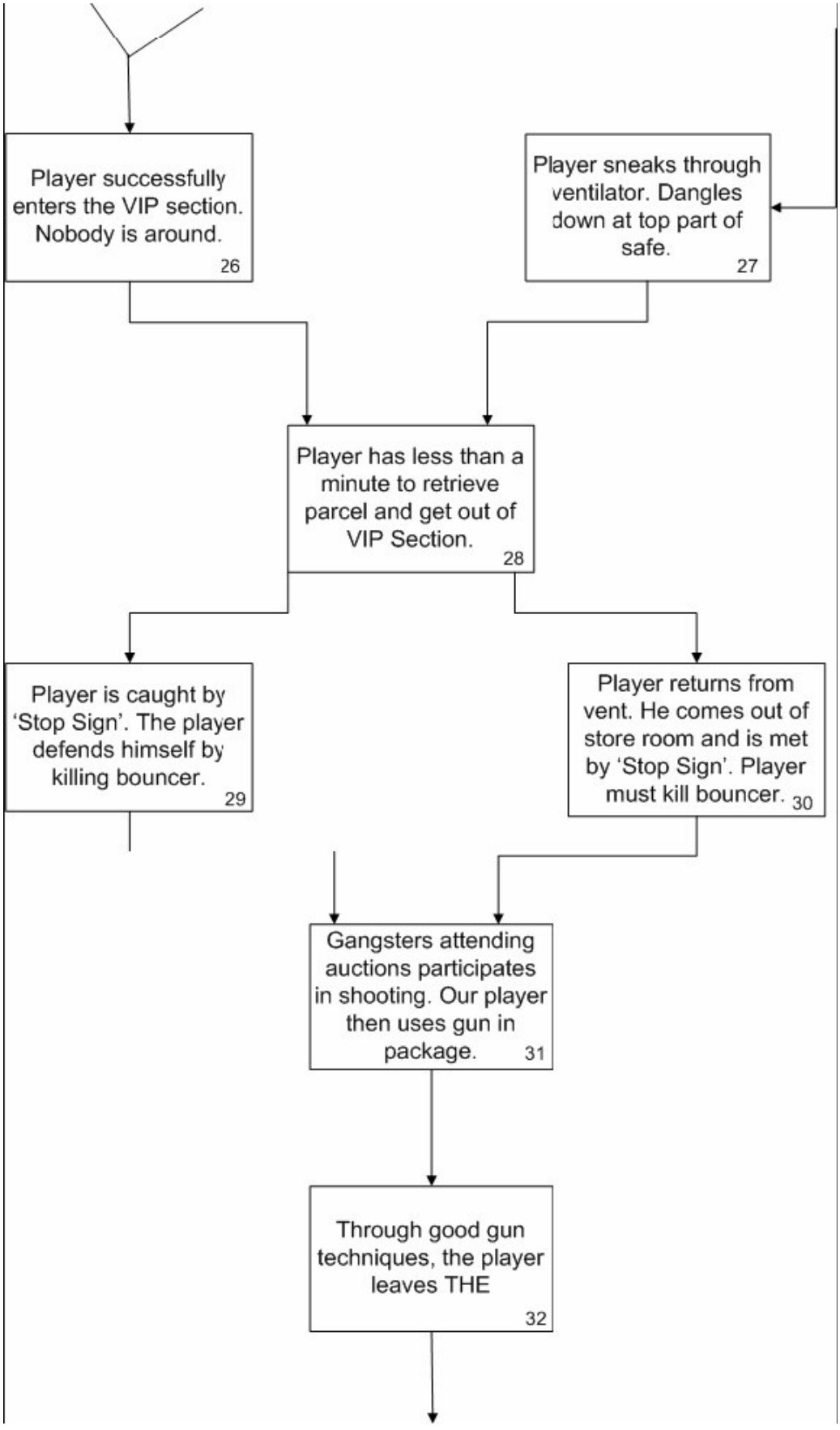
Music

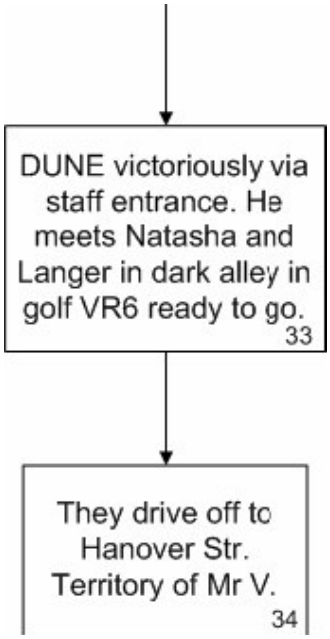
In Mr. V's office the music is dark, atmospheric and 'scummy' with deep baselines and a slow tempo. Music is muffled outside the club; only bass line can be heard. When you go inside the House beat of the club becomes louder and clearer. The further into the club the player gets the more the house beat is audible. On or near the dance-floor the afro-house beat is loud, with Moroccan musical elements. The final part of the scene will be a gun battle the beat morphs into an up-tempo, aggressive beat (Prodigy, etc).

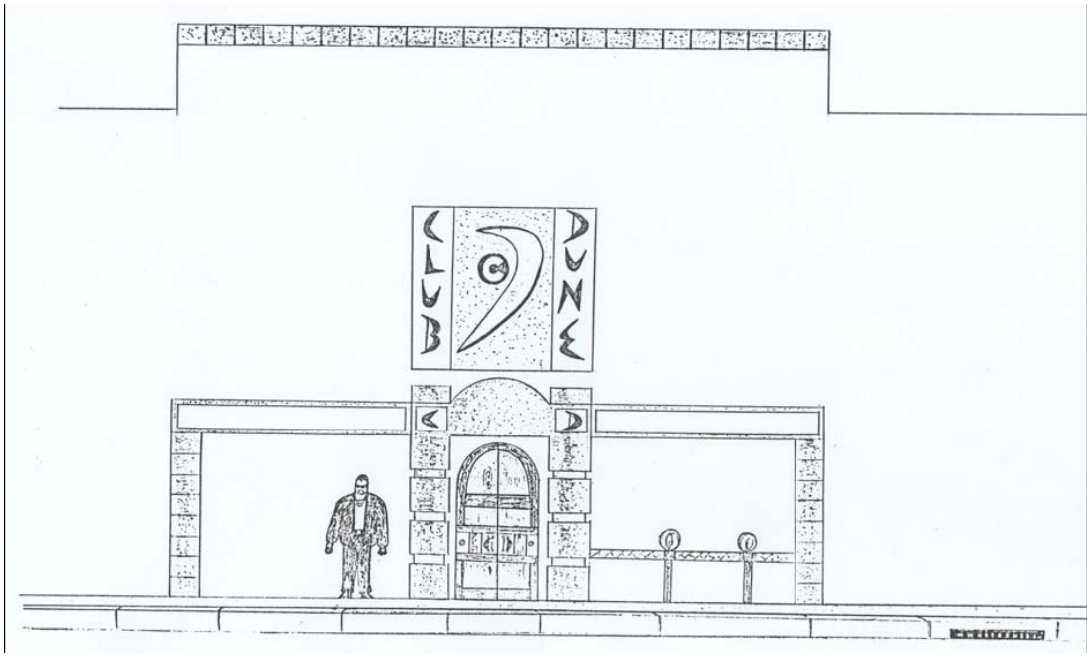

CLUB DUNE
...A FLOW CHART
Mr V receives email of a prototype gun on auction at Club Dune.
1
He gets call from one of his men "Langer" informing him that The Dune is seized by gang "X".
2
The player enters the scene, outside in main road Hanover, mugging/ conning/ a lady tourist from her necklace.
3
Mr V requests Langer to find a suitable person to get in the The Dune and steal the gun before auction time.
4
Langer, calling from outside main road Handover, sees the player mugging lady.
5

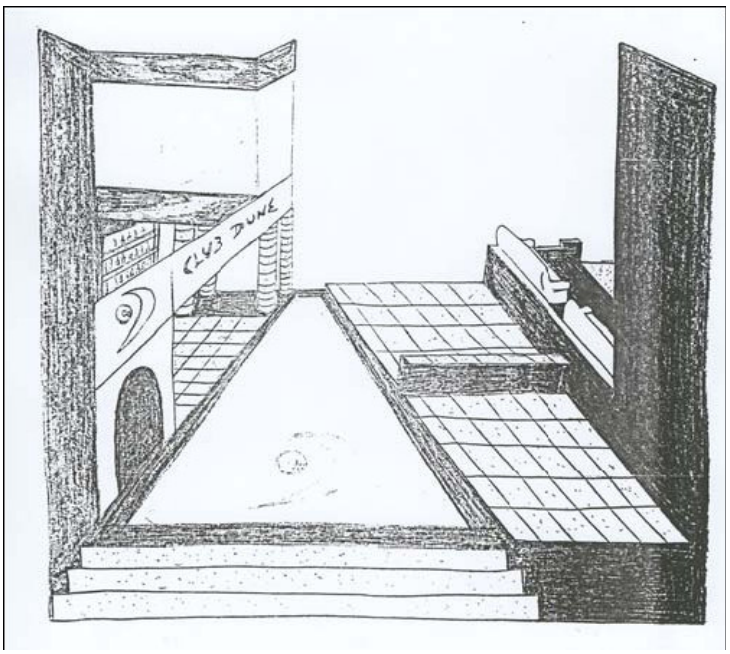

Langer informs Mr V that he has found someone. 6
The player approaches Langer and tries to sell necklace. 7
Instead Langer offers the player a job – ie. To steal "package" at The Dune. Langer gives the player a cell phone to communicate with. 8
The player starts his mission. 9
ENTER THE DUNE 10
FRONT ENTRANCE 11
BACK ENTRANCE 12
Convince the bouncer that you are a part of the auction. 13
Sweet talk two ladies in the queue. In the meantime steal their tickets. 14
Go to side entrance and mug staff member on a smoke break. Take his wrist pass. 15
INSIDE OF THE CLUB 16
Player gets SMS from Langer. 17


Go to bar and ask barkeep for Natasha.
18
Player follows Natasha into store room.
19
Natasha informs player about mission. Follow instructions i.e. steal appropriate gear for VIP.
20
One dressed in VIP gear, meet Natasha in store room.
21
Natasha informs player about next phase. She either gives player...
22
Information to get the pin. Number of the safe and an entry pass.
23
Information on how to get pass “Stop Sign” (the bouncer).
24
Information on how to get to the safe via the vent.
25


Player successfully enters the VIP section. Nobody is around. 26
Player sneaks through ventilator. Dangles down at top part of safe. 27
Player has less than a minute to retrieve parcel and get out of VIP Section. 28
Player is caught by 'Stop Sign'. The player defends himself by killing bouncer. 29
Player returns from vent. He comes out of store room and is met by 'Stop Sign'. Player must kill bouncer. 30
Gangsters attending auctions participates in shooting. Our player then uses gun in package. 31
Through good gun techniques, the player leaves THE 32


DUNE victoriously via staff entrance. He meets Natasha and Langer in dark alley in golf VR6 ready to go.
33
They drive off to Hanover Str. Territory of Mr V.
34


CLUB
DUNE



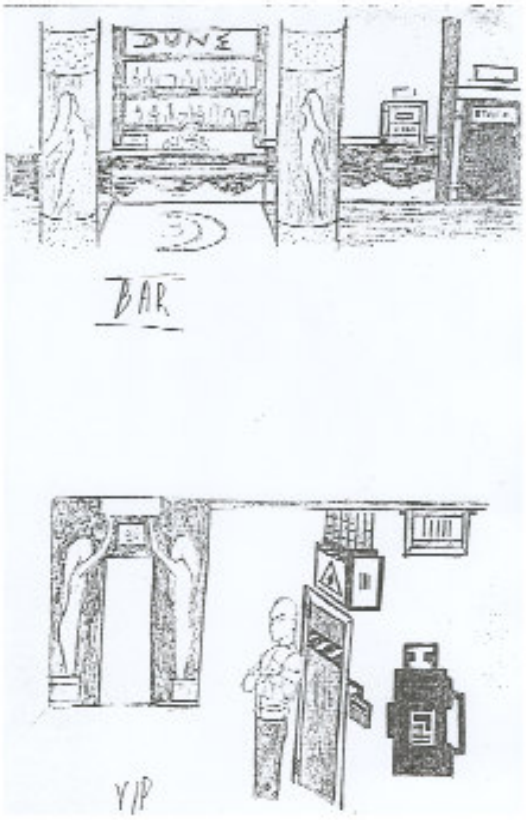


Figure 6

Appendix B: Table of metric for analysing XML files in Chapter 7 Evaluation

This table describes in detail the aspects to consider in evaluating the successful design of VRBridge and the multiple representations. These fall into four categories: VR authoring expertise; Constructivist design values; System and Interface Design; and External Representations

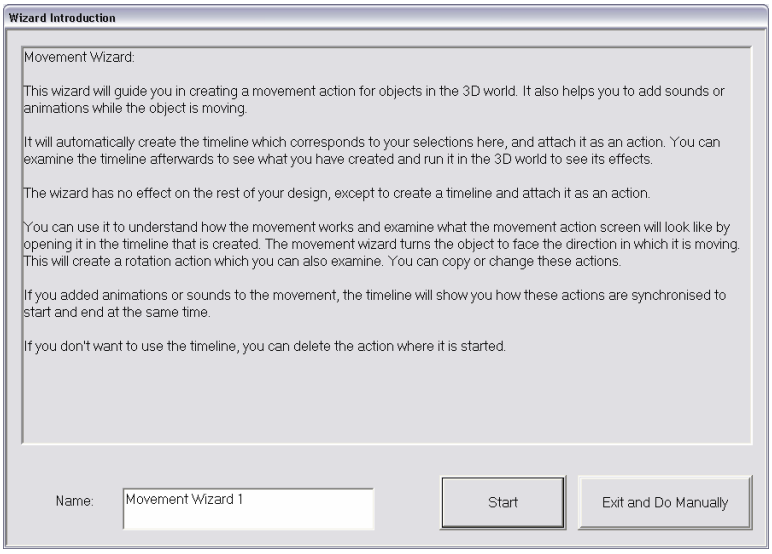
Category	**Aspect**	**Questions**
VR Authoring Expertise	Details	Do users consider relevant interaction details?
	Player freedom	Do users consider alternative actions by the player?
	Non-linear thinking	Do users consider multiple complex non-linear interactions?
	Programming	Do users understand programming constructs and their meaning?
	Time and Space	Do users understand timed and spatial aspects of interactions?
	Composing interactions	Do users understand basic triggers, conditions and actions as building blocks for actions, and how these can be composed?
Constructivist Design Principles	Simplicity	Are VRBridge and the design aids simple to use and understand?
	Multiplicity	Do users benefit from multiplicity and use all information available? Do they work well with multiple windows?
	Exploration	Do users explore VRBridge and the design aids actively and effectively to gain information about the design?
	Control	Do users feel in control of their process and understand how to accomplish tasks without external aid?
	Reflection	Do users reflect on the design and understand domain-related significance? Do the representations foster more reflection?
System and Interface Design	Mapping	Is the mapping between VRBridge and 3D interactions clear?
	Visibility	Are the system status and action alternatives obvious?
	Constraints	Do VRBridge and the representations provide constraints on correct actions?
	Flexibility	Can users work in flexible ways with the system?
	Error-handing	Do the Triggersets and representations make errors easy to find?
	Feedback	Does the system provide feedback about authoring?
	Terminology	Do users understand the terminology and its domain meaning?
Effective	Distributed	Do the representations help with *lookahead* and correct biases for

Category	**Aspect**	**Questions**
External Representations	cognition	action?
	Cognitive Offloading	Do the representations provide help with working memory, perception of details, and structure and pattern recognition?
	Re-representation	Do the representations assist in abstraction? Is less reformulation required as aspects are described explicitly?
	Constraints	Do the representations provide effective temporal and spatial constraints and aid for spatial structuring.
	Floorplan	Does the Floorplan help users in spatial understanding, and considering implications of perceptual opportunities of VE and effects of Triggerset actions?
	Sequence Diagram	Does the Sequence Diagram help users think non-linearly and understand how interactions connect? Does the network formalism help users understand structure and follow narrative paths?
	Timelines	Do the Timelines help with temporal understanding? Can users understand better how interactions work together in time?

Appendix C: Wizards Created for VRBridge

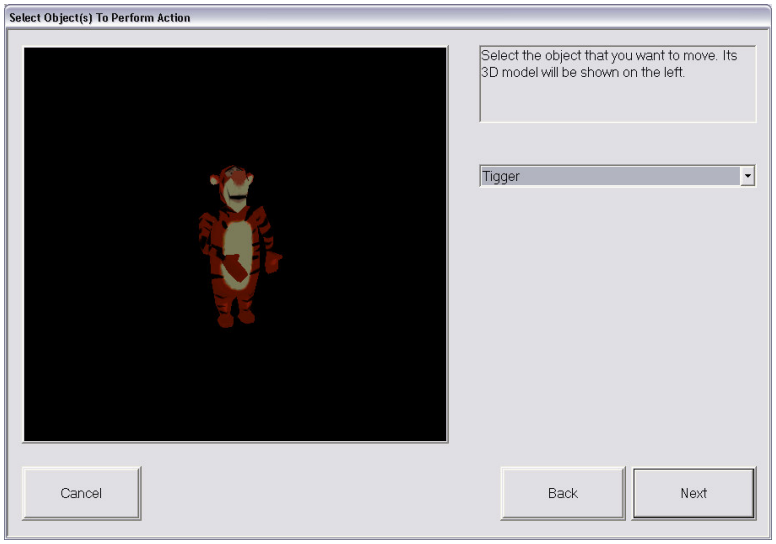
Appendix C1: Movement Wizard

This appendix describes the Movement Wizard of VRBridge, with screenshots. When it is selected, it opens the following screen:

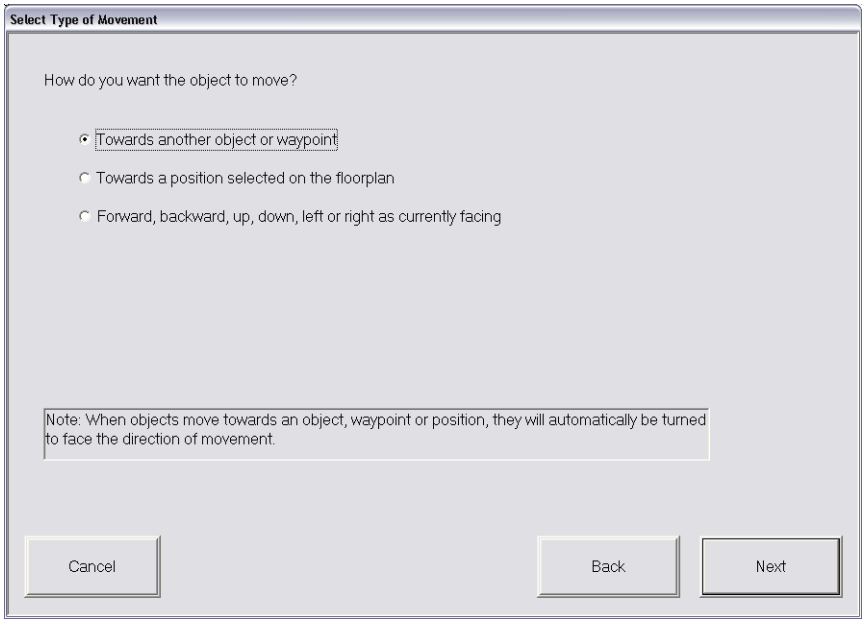


The user provides the wizard with a name, used to name the resulting Timeline. On each screen, the user can return to previous screens and edit them. The user presses *Start* to begin.

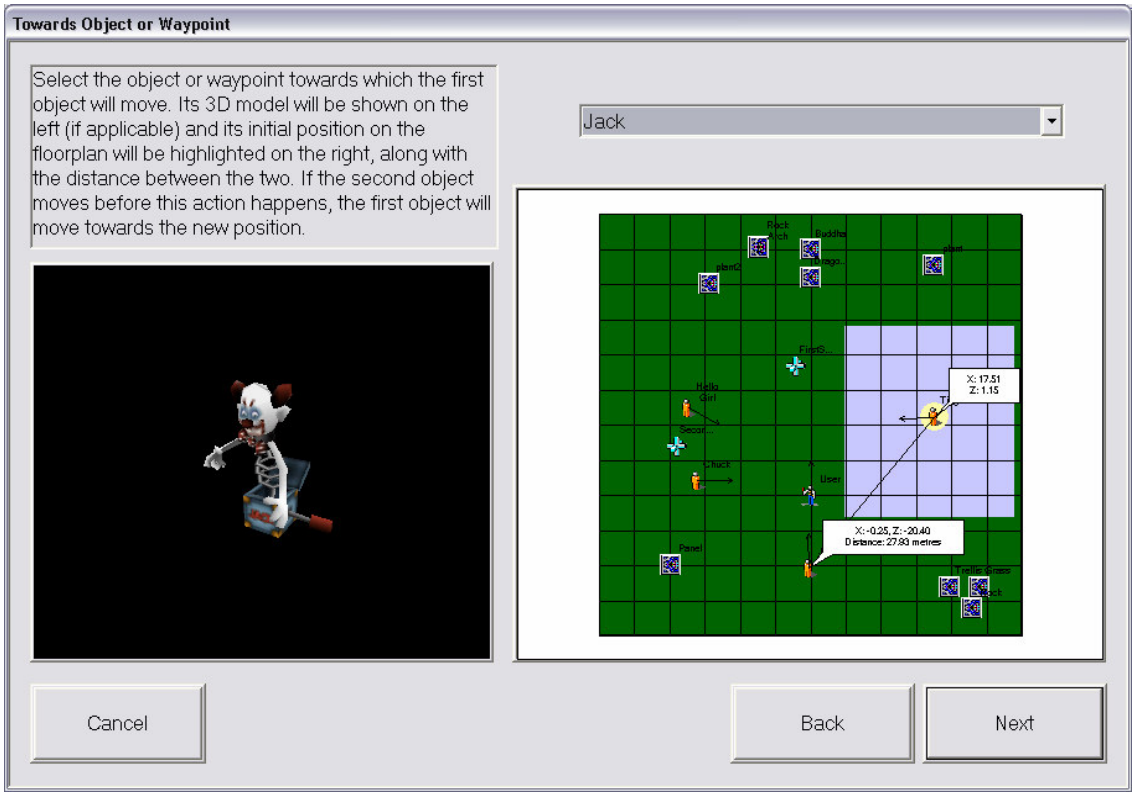
1. First the user selects the object which will perform the movement from a drop down list, with a preview.



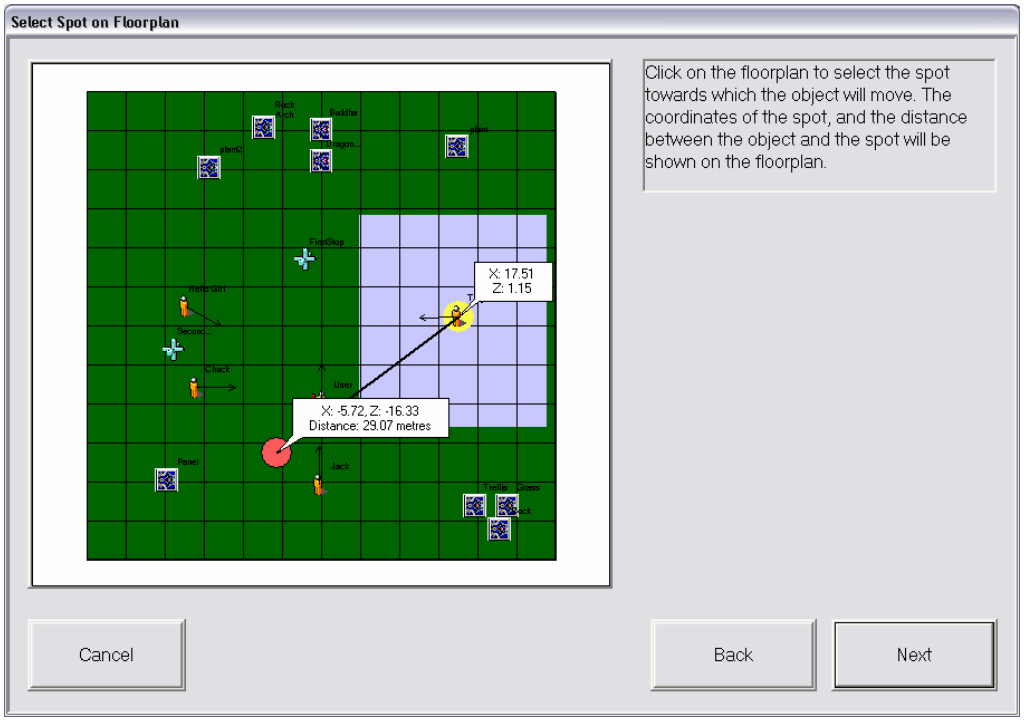
2. Second, the user selects the type of movement to perform.



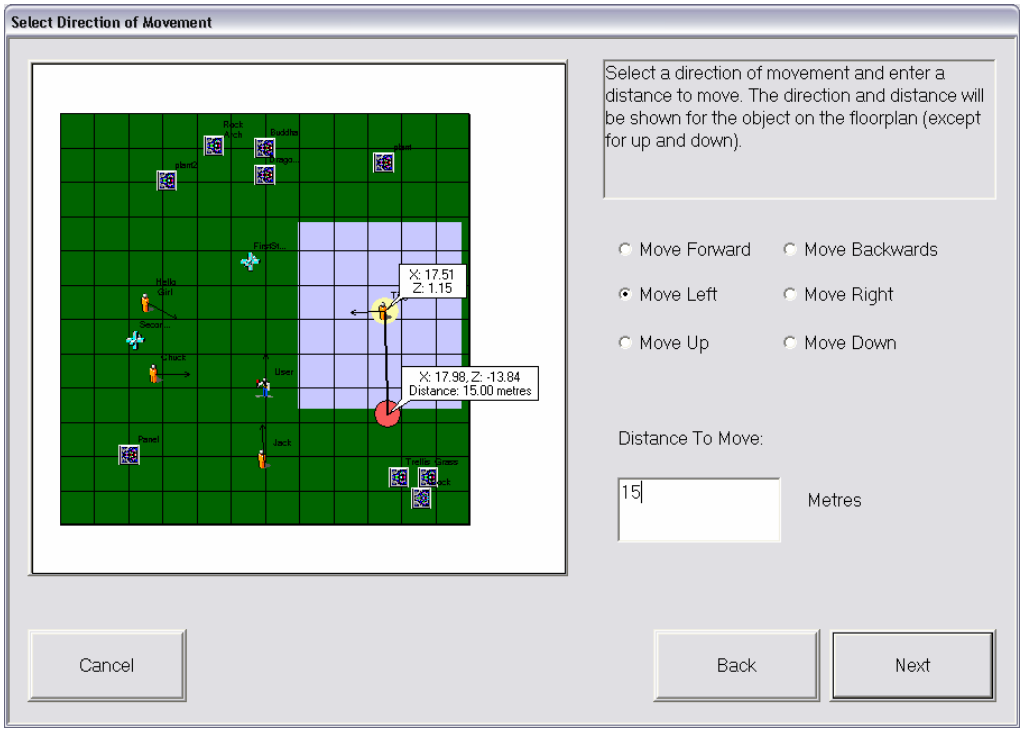
3. Depending on the option selected in (2), three different screens open.
   a. Movement towards another object or waypoint.



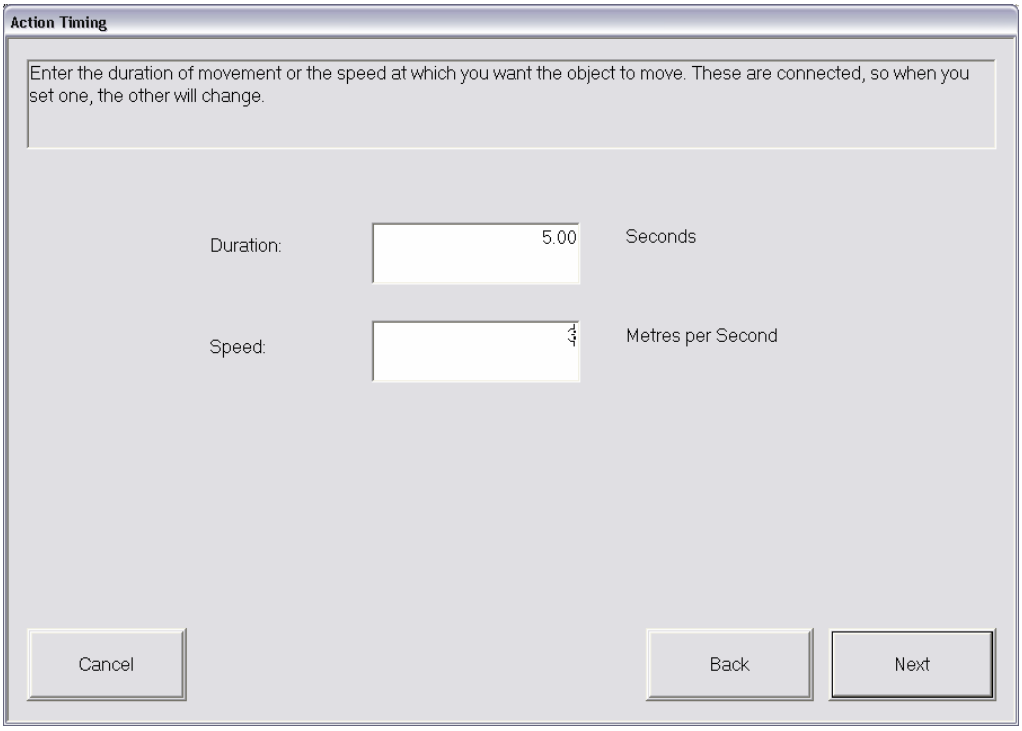
b. Movement towards a position selected on the Floorplan.



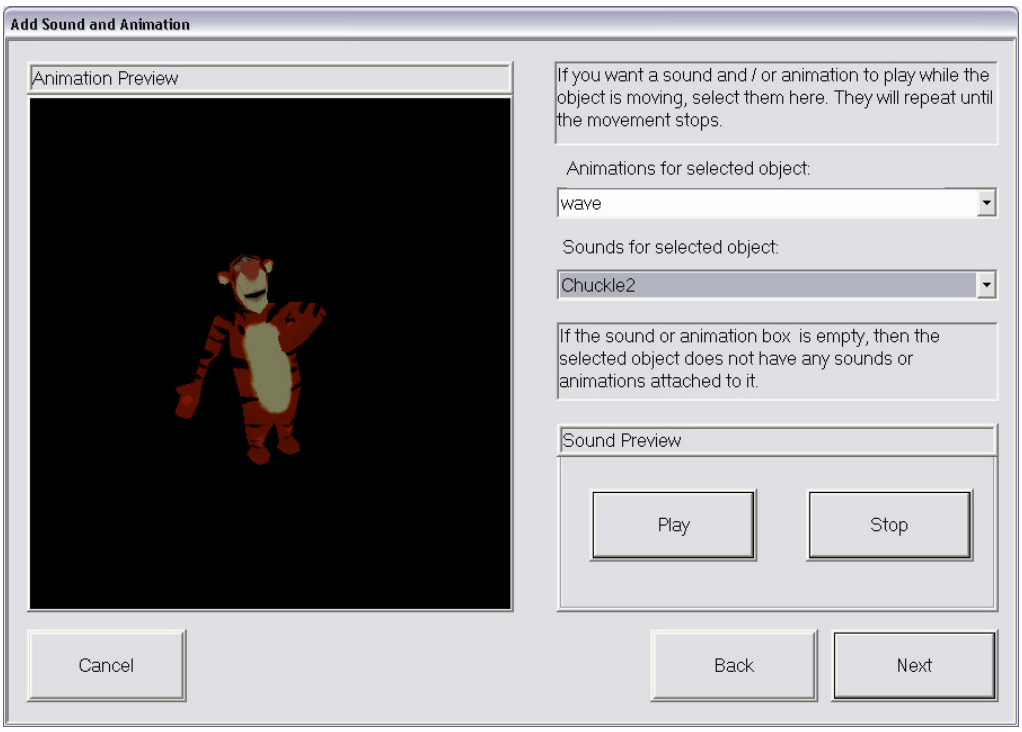
c. Movement in a direction.



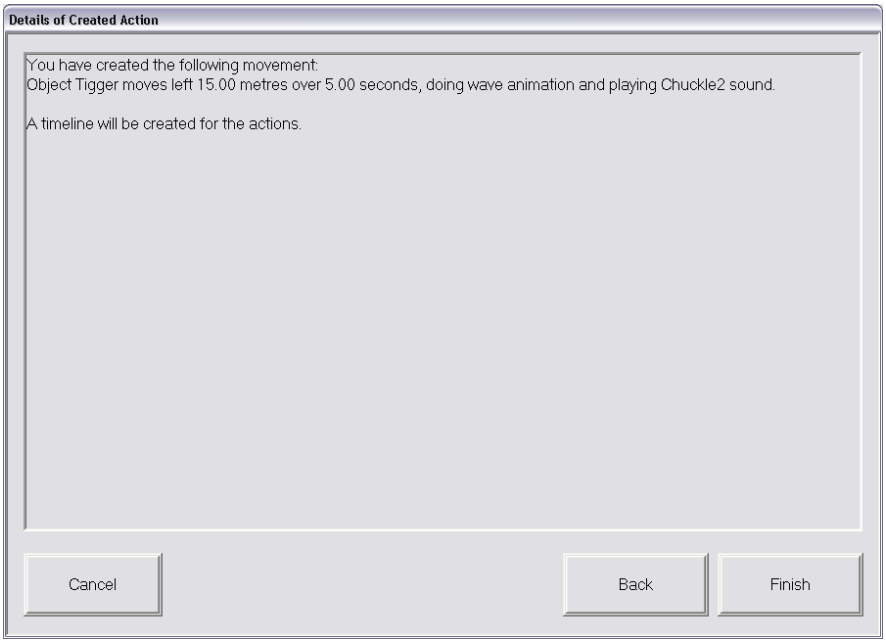
4. Once the type of movement has been selected, a screen titled *Action Timing* opens.



5. This is followed by a screen for adding animations and sounds to the movement.

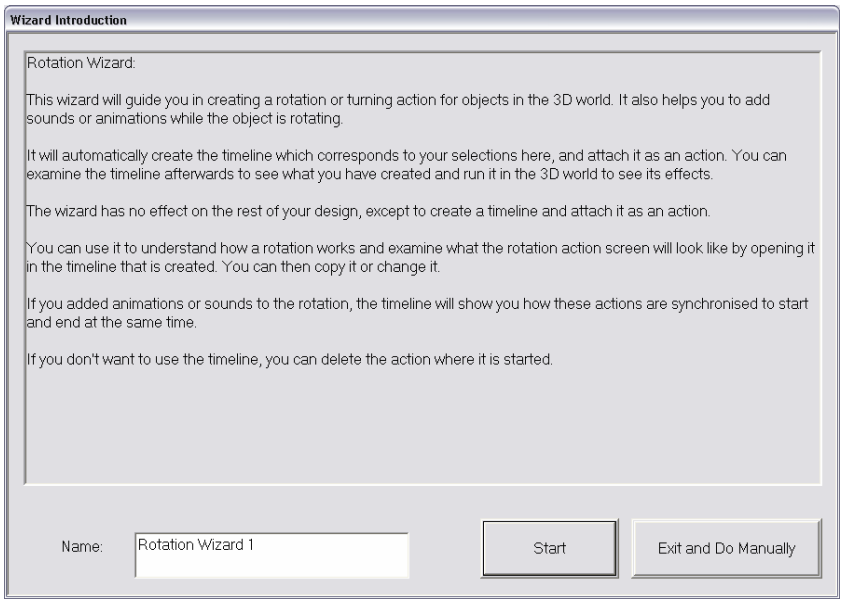


6. When the user has finished creating the movement, a screen appears which summarises the results.



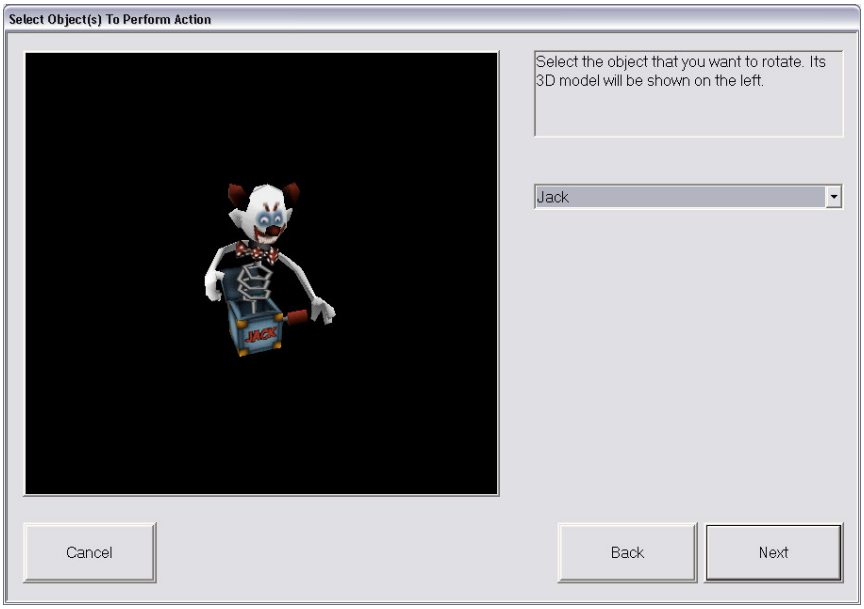
Appendix C2: Rotation Wizard

This appendix describes the Rotation Wizard of VRBridge, with screenshots. When it is selected, it opens the following screen:

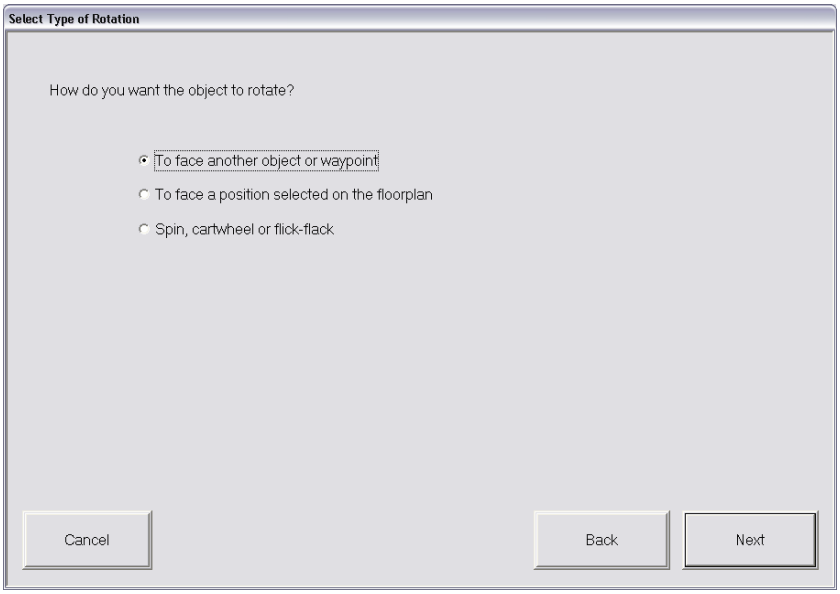


The format of the rotation wizard is very similar to that of the movement wizard.

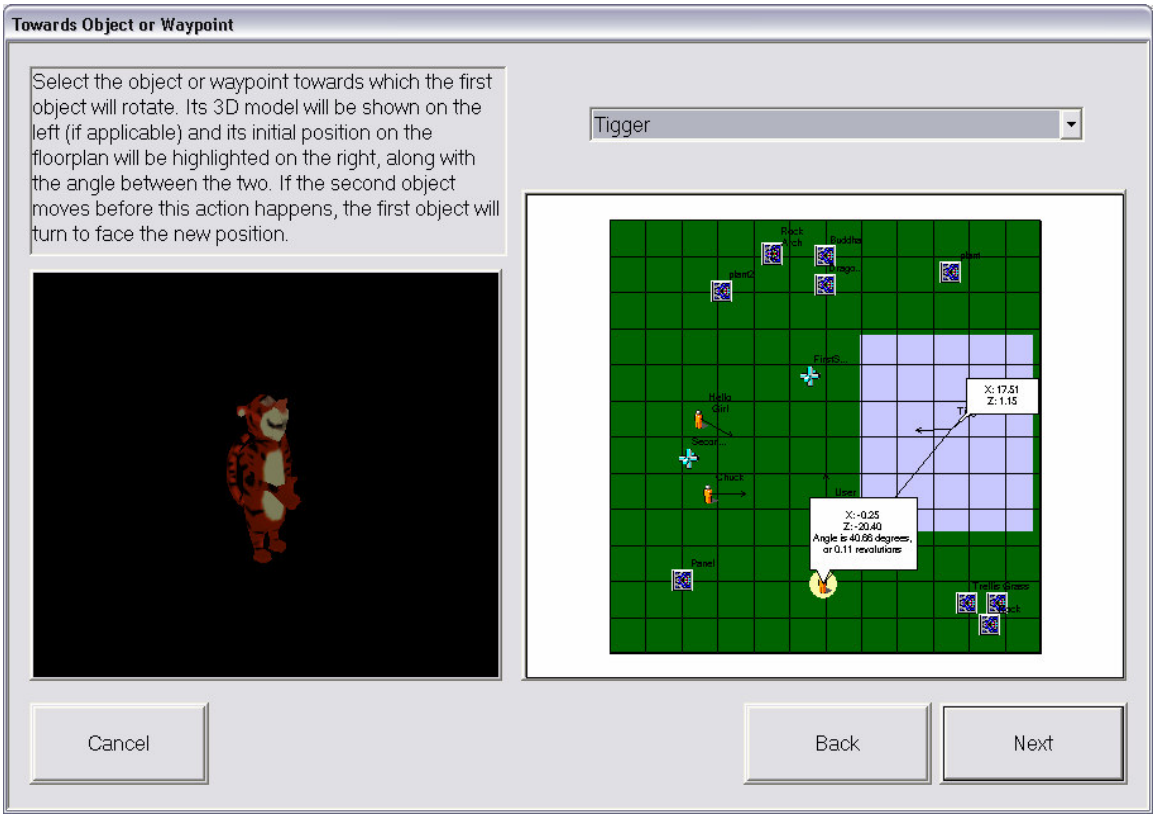
1. First the user selects the object which will perform the rotation from a drop down list.



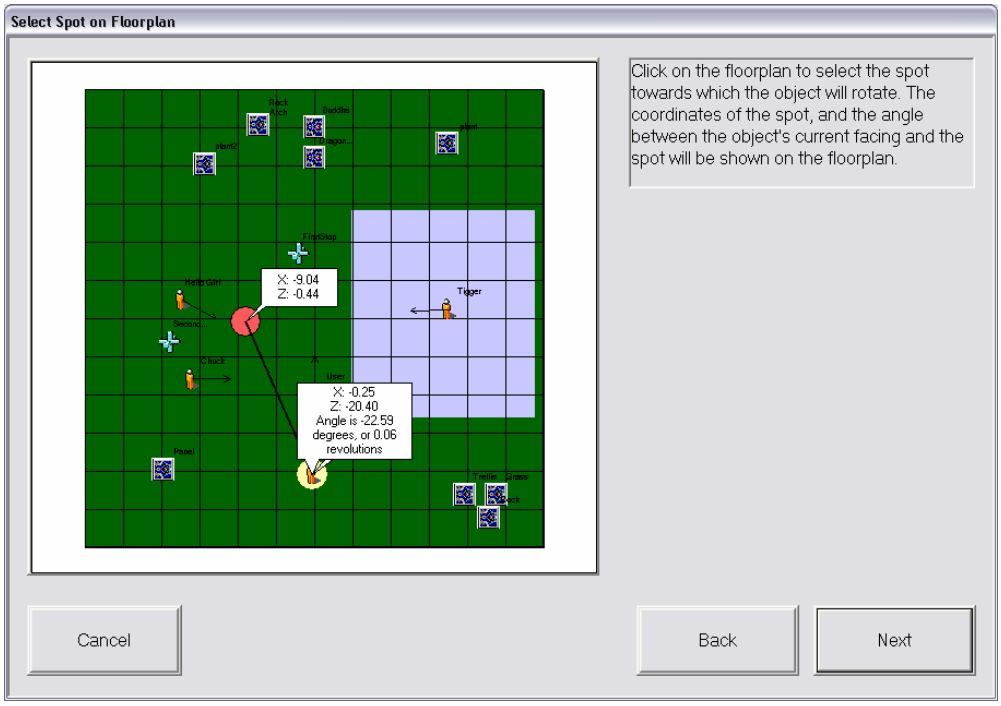
2. Second, the user selects the type of rotation to perform.



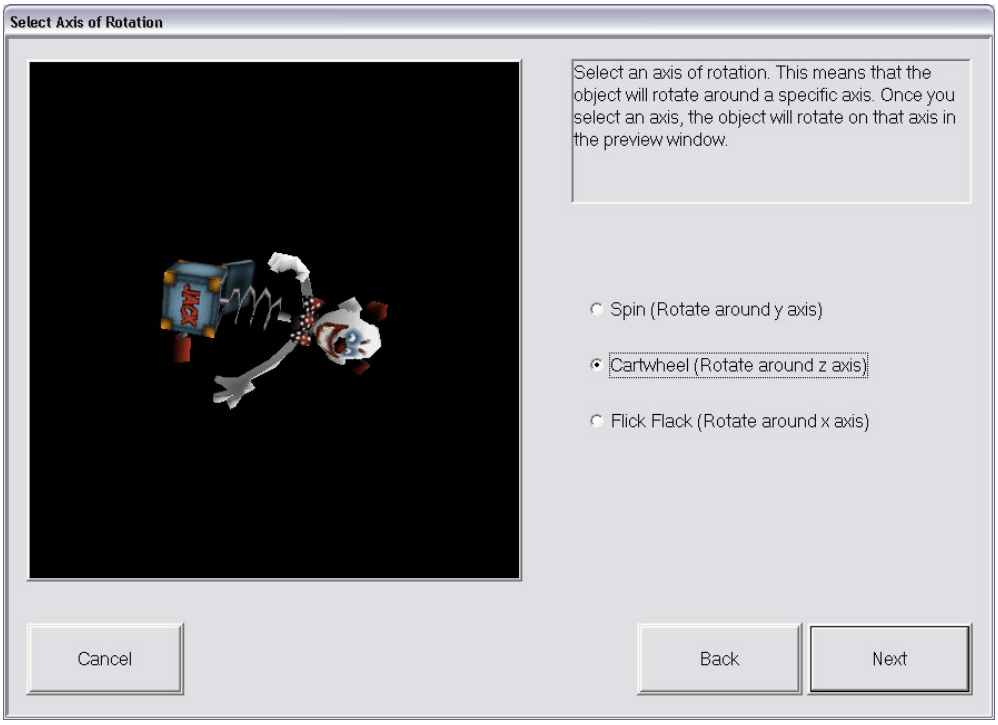
3. Depending on the option which is selected in (2), three different screens open.
   a. Rotating towards another object or waypoint.

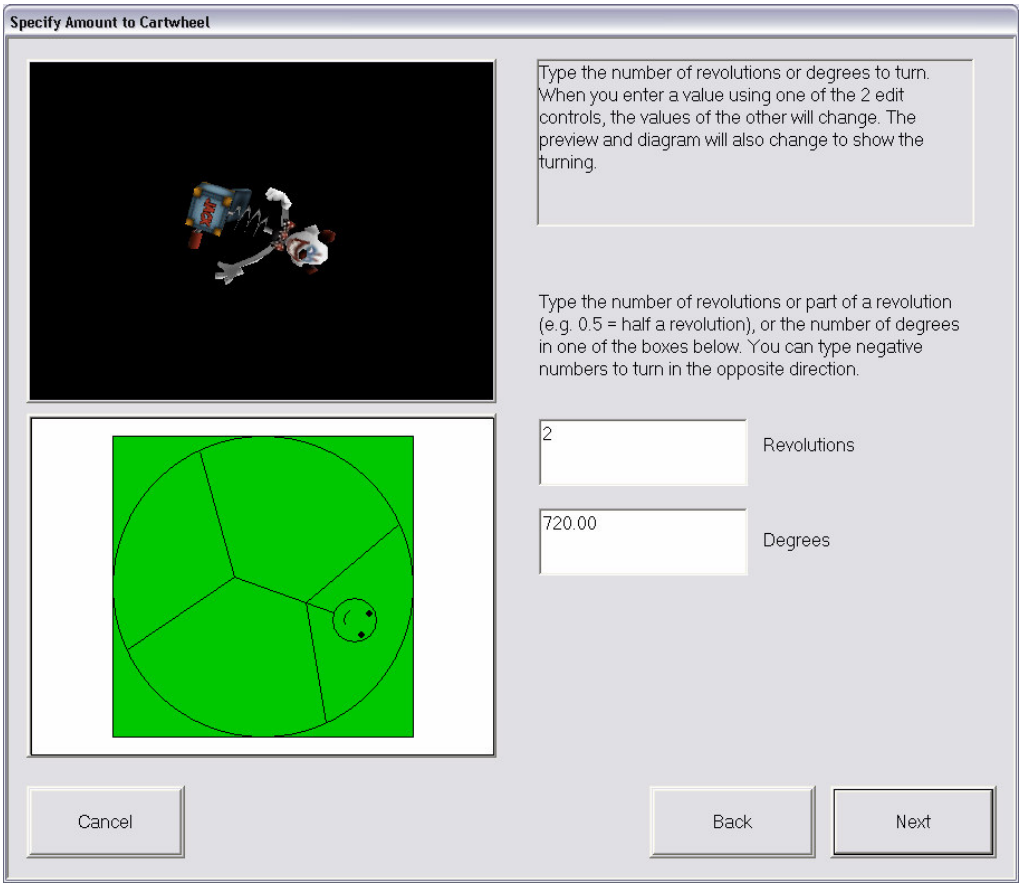


b. Rotation towards a position selected on the Floorplan

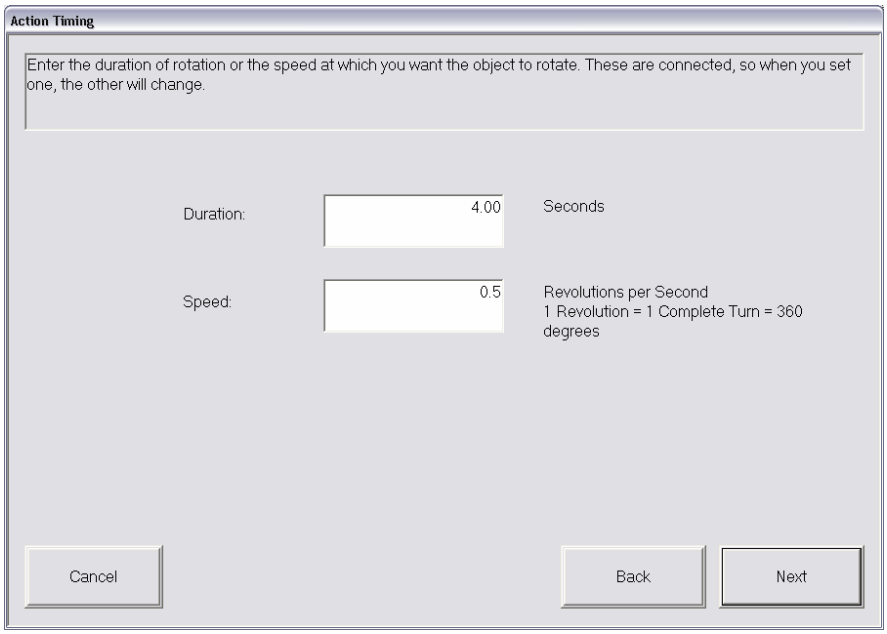


c. Rotation around an axis (this option leads to two connected screens)

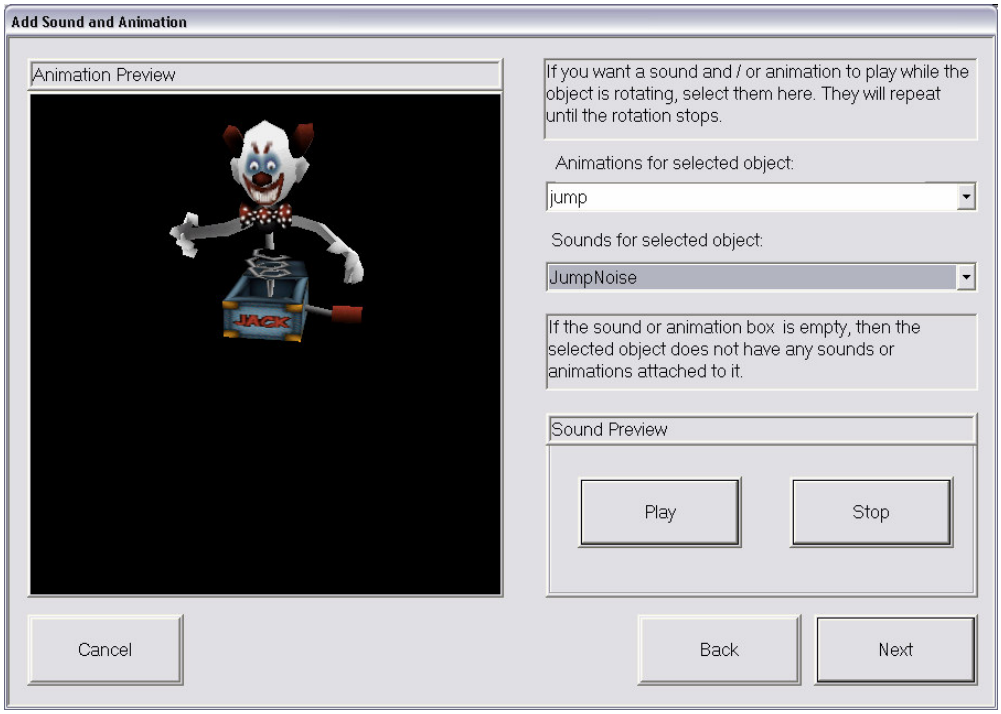




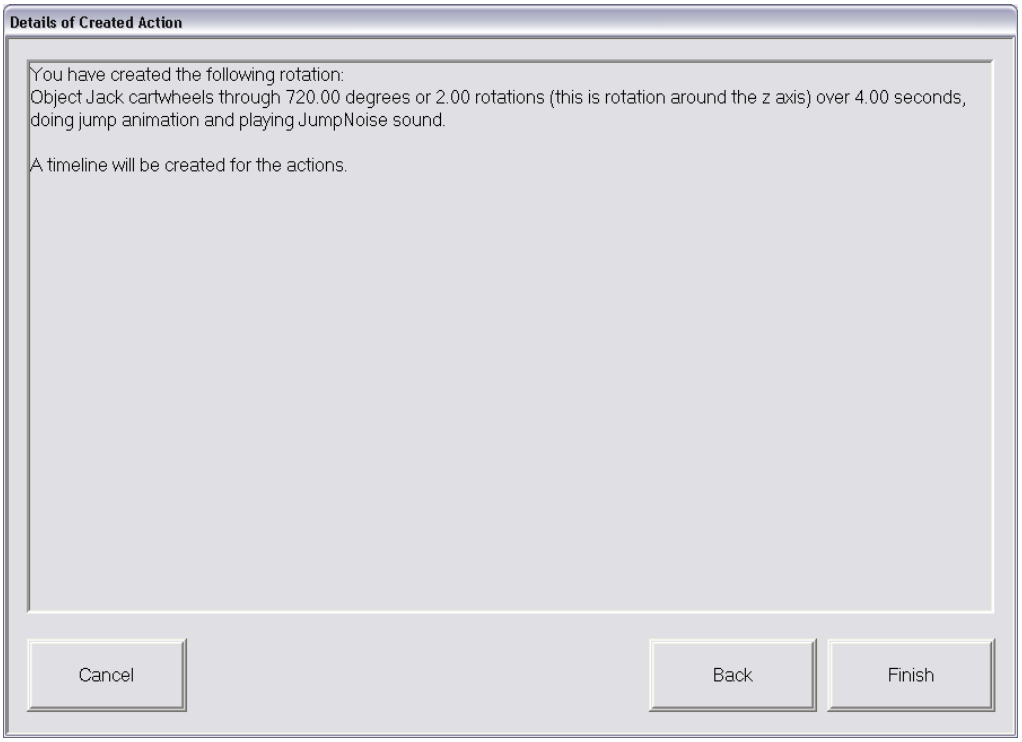
4. Once the type of rotation has been selected, a screen titled *Action Timing* opens.



5. As with the movement wizard, a screen appears for adding sounds and animation.



6. When the user has finished creating the rotation, a final screen appears which summarises the results.



Appendix D: Table of metric for analysing XML files in final evaluation of VRBridge

Metric Item	Description	Value
Common Section		
Inevitable Sequencing	Timelines and Timeline sequencing	Score ranges from 0 for no Timelines created to 10 for Timelines with multiple objects and synchronised actions which are added meaningfully to the Triggersets, and that interact with other Triggersets (e.g., action in Timeline triggers further interactions): 5 for Timelines and 5 for sequencing. *Timelines*: 0 – no Timelines created, 1 – created a Timeline, 2 – used Timelines in Triggersets, 3 – included multiple actions on Timeline, 4 – included multiple objects and actions on Timeline, 5 – objects and actions well synchronised. *Sequencing*: 0 – no Timelines added to Triggersets, 1 – Timelines added to Triggersets, 2 – Timelines added meaningfully, 3 – actions on Timeline synchronised well, 4 – actions on Timeline synchronised and Timeline combines well with Triggersets, 5 – Timeline interacts with other Triggersets (e.g. action in Timeline triggers further interactions).
Variable Sequencing	Triggers, actions and conditions; and Triggerset sequencing	Score ranges from 0 for no Triggersets created to 20 for multiple Triggersets with different kinds of triggers, conditions and actions that are linked, synchronised and appropriate: 5 for triggers, 5 for actions, 5 for conditions and 5 for sequencing. *Triggers*: 0 – no Triggersets, 1 – created Triggersets that do not work (e.g. do not execute, have unintentional side-effects), 2 – some but not all Triggersets work, 3 – multiple Triggersets all work, 4 – different kinds of triggers used, 5 – triggers are linked and interesting (e.g. different kinds of triggers used appropriately). *Actions*: 0 – no actions, 1 – some actions created, 2 – single action per Triggerset, 3 – multiple actions appropriately in Triggerset, 4 – actions

Metric Item	Description	Value
		are customised (e.g. actions synchronised by making lengths equal), 5 – actions customised successfully and an action is used as a trigger. *Conditions*: 0 – conditions added which do not work (i.e. make the Triggerset impossible), 1 – no conditions added, 2 – conditions added and work, 3 – conditions are appropriately to context, 4 – conditions create interesting interactions (e.g. making sure action only happens after another because of spatial conditions), 5 – used to create action alternatives depending on conditions. *Sequencing*: 0 – no working Triggersets, 1 – single Triggerset, 2 – multiple Triggersets which are not linked to each other, 3 – some Triggersets are linked, 4 – more than two Triggersets linked in a sequence, 5 – more than two Triggersets linked and synchronised in appropriate sequences
Features	Usage of system features	Score ranges from 0 for no features explored to 5 for multiple features used meaningfully and in interesting ways (e.g., waypoints used to mark a route, starting positions of objects customised). 1 – features explored but not used (e.g. add waypoint or location), 2 – single feature used in Triggersets or Timelines, 3 – multiple features used in Triggersets or Timelines, 4 – features used meaningfully (e.g. change player speed on object panel, change initial object position for better starting configuration or waypoint placed in effective position), 5 – features used in interesting ways (e.g. locations used for condition alternatives, or waypoints used to mark a route).
Creativity	Interesting design, usage of VE space, and appropriate and natural looking interactions	Score ranges from 0 for no interactions to 15 for a VE that is interesting (i.e., have created an interesting story from the task description), where objects move and orient appropriately in the space (e.g., objects do not collide, triggers set to appropriate distances) and look natural (e.g., appropriate synchronised animations and sounds, correct facing): 5 for interesting design, 5 for usage of space and 5 for natural interactions.

Metric Item	Description	Value
		Interesting design: 0 – nothing happens, 1 – show interest in task, 2 – basic task well attempted, 3 – more than the basic task has been attempted with some success, 4 – contains exciting or out-of-the-ordinary elements (e.g. have played with Triggerset effects, created an interesting story from the task description), 5 – the whole task is put together in an interesting and exciting manner. *Space*: 0 – no movement of objects, 1 – objects move, 2 – entire VE space used, 3 – show understanding of space and size of VE (e.g. objects move appropriate distances), 4 – objects move appropriately and do not collide, 5 – all movement and orientation in space is correct (e.g. objects move into appropriate distance and facing for intimate and distant interactions, triggers set to appropriate distances) *Naturalness*: 0 – nothing happens, 1 – basic interactions attempted successfully, 2 – basic attempts have been made to look natural (e.g. appropriate animations and sounds chosen, actions can be understood as part of a narrative), 3 – some detailed attempts to look natural (e.g. turn to face movement, add synchronised animation while object moves), 4 – basic and detailed attempts to look natural are successful, 5 – entire VE and task looks natural (e.g. all movements have synchronised actions and all objects face correctly)
Player Freedom	Include the player appropriately and allow freedom of movement	Score ranges from 0 where the Triggerset actions are used to move the player, to 10 for multiple forms of player interaction (e.g., Player input, player proximity) added to triggers, conditions and actions (e.g., objects turn to face player during action) which successfully relate to the narrative of the VE. 0 – use Triggerset actions to move end user, 1 – move player in actions but could be justified by narrative, 2 – no explicit player interactions, 3 – move player in actions but well justified by narrative, 4 – user input trigger added, 5 – user input trigger(s) added with some connection to task narrative, 6 – user input trigger added which successfully

Metric Item	Description	Value
		connects to narrative (e.g. user presses key when object is nearby for interaction with object), 7 – player interaction other than user input added (e.g. via proximity or viewpoint of user), 8 – player proximity or viewpoint used as trigger with some connection to task narrative, 9 – multiple forms of player interaction added in triggers which successfully relation to narrative, 10 – multiple forms of player interaction added to triggers, conditions and actions (e.g. objects turn to face user during action)
Common Section Total		**60**
	Task 1	
Trigger Song	Start Tigger's song	Score ranges from 0 for nothing added to 4 if the song is added naturally and appropriate to the narrative. 1 – add Tigger sound, 2 – add Tigger song, 3 – add song with some attempt at narrative, 4 – add song creatively and naturally
Tigger Actions	Provide 3 actions for Tigger	Score ranges from 0 for no actions to 16 for all actions added, which are appropriate to the story and synchronised with the song and other actions: 4 for each action, and 4 if actions happen during song *For each action*: 1 – add action, 2 – add appropriate action, 3 – customise and synchronise action to some extent, 4 – appropriate action synchronised with song and other actions *For during song*: 1 – one action during song, 2 – two actions during song, 3 – all actions during song, 4 – actions synchronised with each other and well timed during song
Other Character Actions	Provide 2 actions for 2 other characters	Score ranges from 0 for no actions to 20 for narratively appropriate and synchronised actions played by 2 characters during the song: 4 for each action, and 4 if actions happen during song These are scored in the same way as those for Tigger.
	Task 2	
Patrol	Attempt to make a character patrol the area	Score ranges from 0 for no movement to 8 for the patrol fully accomplished in a narratively interesting and coherent manner.

Metric Item	Description	Value
		1 – character moves, 2 – character moves with apparent objective, 3 – character moves around space, 4 – shows attempts to make patrol, 5 – character moves around space and performs actions, 6 – patrol partially accomplished, 7 – patrol fully accomplished, 8 – patrol accomplished in narratively interesting and coherent manner.
Return to Start	Make character return to where he started	Score ranges from 0 for no return movement to 4 if the character returns to where he started. 1 – moves somewhat in direction of start, 2 – moves obviously in direction of start, 3 – returns to general area of start, 4 – returns to start
Two Stops	Do 2 stops on patrol, one with another character	Score ranges from 0 for no movement to 12 if both stops accomplished, are narratively appropriate: 4 for each stop, and 4 if one stop is with a character *For each stop*: 1 – moves towards new spot, 2 – stops at new spot, 3 – shows attempts at patrol-appropriate stops, 4 – stop is narratively appropriate *For character stop*: 1 – approaches character, 2 – stops at character, 3 – stops at character and interacts, 4 – stops and interacts in narratively interesting way
Action at Stops	Do an action at each stop	Score ranges from 0 for no actions to 8 for actions at both stops which are narratively interesting and appropriate to the patrol: 4 for each action *For each action*: 1 – patroller does an action, 2 – patroller does action well, 3 – some attempt at narratively interesting action, 4 – action narratively interesting and appropriate to patrol
Actor Response	Second character responds to patroller	Score ranges from 0 for no action to 4 for an action which is synchronised with the patroller action and narratively appropriate. 1 – character performs action, 2 – character performs appropriate action well, 3 – character action synchronised with patroller action, 4 – character action related to narrative and patroller action

Metric Item	Description	Value
Repeat Patrol	Repeat once return to start	Score ranges from 0 for no patrol to 4 for repeating the patrol correctly. 1 – does full patrol returning to start position, 2 – show attempts to repeat patrol, 3 – repeat part of patrol, 4 – repeat patrol
Task 3		
Greeting	Character greets player	Score ranges from 0 for no acknowledgement of player to 5 where the character approaches the player, greets the player and returns to the starting area. 1 – character acknowledges player, 2 – character moves up to player, 3 – character moves up and acknowledges player, 4 – character also returns to sound system area, 5 – actions appropriate for greeting
Split Action	Action is split based on player actions	Score ranges from 0 for no split to 8 if the action is successfully split and is narratively interesting. 1 – does both split actions, 2 – does both split actions well, 3 – shows attempts to split the action, 4 – shows attempts to split action and partially succeeds, 5 – succeeds at splitting action only if player timing is exact, 6 – succeeds at splitting action, 7 – succeeds at splitting action and attempts to make narratively interesting, 8 – succeeds at splitting action and is narratively interesting
Nice Response	Characters say nice things to player	Score ranges from 0 for no response to 4 where multiple characters say appropriate nice things and the actions are well synchronised. 1 – attempt made, 2 – multiple characters say nice things, 3 – characters say appropriate nice things, 4 – characters say appropriate nice things and timing is good
Nasty Response	Characters say nasty things to player	Score ranges from 0 for no response to 4 where multiple characters say appropriate nasty things and the actions are well synchronised. 1 – attempt made, 2 – multiple characters say nasty things, 3 – characters say appropriate nasty things, 4 – characters say appropriate

Metric Item	Description	Value
		nasty things and timing is good
Turn on Music	Character turns on music, i.e. does appropriate action near sound system	Score ranges from 0 for no music to 5 where music plays and the timing and narrative of the action are good. 1 – music is played, 2 – character moves to sound system and music plays, 3 – character does action near sound system and music plays, 4 – attempts made to do appropriate action, 5 – timing and narrative of action to turn on music are good
Dance	Characters dance to music i.e. do appropriate actions when music plays	Score ranges from 0 for no movement to 14 for all the characters dancing with each other, where the movements simulate dancing well, and the dancing is varied and timed with the music: 10 for dancing and 4 if multiple characters dance *For dancing*: 1 – characters move, 2 – characters move during music, 3 – characters do multiple or repeated movement, 4 – basic attempts made to simulate dancing, 5 – some interesting attempts made, 6 – characters dance, 7 – dancing varied (i.e. do not just repeat one animation), 8 – dancing varied and timed with music (e.g. repeated for duration, appropriate to music choice), 9 – attempts to make characters dance with each other, 10 – characters dance with each other *For multiple characters*: 1 – single character dances, 2 – 2 characters dance, 3 – half or more characters dance, 4 – all characters dance
Task Total		**40**
Total		**100**

Appendix E: Questionnaires devised for final evaluation of VRBridge

Appendix E1: Demographic Questionnaire

Age:	
Gender:	
Department:	
Level of Study:	
Can you program in a computer language (E.g. c++, Java, c#)?	**Yes** **No**

Indicate your familiarity with the following computer software and systems by ticking in the appropriate column.

Device	**Frequency of use**		
	Never	**Seldom**	**Often**
Desktop computer			
Lap top computer			
Word processor			
Spreadsheet software			
Computer games			
First person 3D computer games			
Email			
Internet			

Appendix E2: System Questionnaire

1	Triggersets were	confusing	clear
		1 2 3 4 5 6 7 8 9	
		difficult to use	easy to use
		1 2 3 4 5 6 7 8 9	
		frustrating	fun
		1 2 3 4 5 6 7 8 9	
2	The power of triggersets to describe actions was	inadequate	adequate
		1 2 3 4 5 6 7 8 9	
3	The effects of triggersets were	confusing	predictable
		1 2 3 4 5 6 7 8 9	
4	The object panel was	confusing	clear
		1 2 3 4 5 6 7 8 9	
		difficult to use	easy to use
		1 2 3 4 5 6 7 8 9	
5	The animation preview was	useless	helpful
		1 2 3 4 5 6 7 8 9	
6	The sound preview was	useless	helpful
		1 2 3 4 5 6 7 8 9	
7	The model preview was	useless	helpful
		1 2 3 4 5 6 7 8 9	
8	Creating locations was	difficult	easy
		1 2 3 4 5 6 7 8 9	
9	Creating waypoints was	difficult	easy
		1 2 3 4 5 6 7 8 9	
10	Viewing locations and waypoints in the 3D world was	useless	helpful
		1 2 3 4 5 6 7 8 9	
11	Working with the 3D world was	difficult	easy
		1 2 3 4 5 6 7 8 9	
12	Timelines were	frustrating	fun
		1 2 3 4 5 6 7 8 9	
		useless	helpful
		1 2 3 4 5 6 7 8 9	
		difficult to use	easy to use
		1 2 3 4 5 6 7 8 9	

	confusing clear 1 2 3 4 5 6 7 8 9

Appendix E3: Representations Questionnaire

1	In general, the diagrams and run mode were	confusing clear 1 2 3 4 5 6 7 8 9
		useless helpful 1 2 3 4 5 6 7 8 9
		frustrating fun 1 2 3 4 5 6 7 8 9
2	The floorplan was	useless helpful 1 2 3 4 5 6 7 8 9
		difficult to use easy to use 1 2 3 4 5 6 7 8 9
		frustrating fun 1 2 3 4 5 6 7 8 9
		confusing clear 1 2 3 4 5 6 7 8 9
3	The amount of manipulation I could do on the floorplan was	inadequate adequate 1 2 3 4 5 6 7 8 9
4	The sequence diagram was	useless helpful 1 2 3 4 5 6 7 8 9
		difficult to use easy to use 1 2 3 4 5 6 7 8 9
		confusing clear 1 2 3 4 5 6 7 8 9
		frustrating fun 1 2 3 4 5 6 7 8 9
5	The amount of manipulation I could do on the sequence was	inadequate adequate 1 2 3 4 5 6 7 8 9
6	The Run Mode was	useless helpful 1 2 3 4 5 6 7 8 9
		confusing clear 1 2 3 4 5 6 7 8 9
		difficult to use easy to use 1 2 3 4 5 6 7 8 9

Appendix E4: Scaffolding Questionnaire

PART 1: General, Context Sensitive Help and Tooltips

1.1. In general help is	confusing	clear
	1 2 3 4 5 6 7 8 9	
1.1.1 The terminology used in the help	confusing	clear
	1 2 3 4 5 6 7 8 9	
1.1.2 Amount of help given	inadequate	adequate
	1 2 3 4 5 6 7 8 9	
1.1.3 Content of help	confusing	clear
	1 2 3 4 5 6 7 8 9	
1.1.4 Help defines specific aspects of the system	inadequately	adequately
	1 2 3 4 5 6 7 8 9	
1.2 Tooltips are	inadequate	adequate
	1 2 3 4 5 6 7 8 9	
	useless	helpful
	1 2 3 4 5 6 7 8 9	
1.3 Context sensitive help is	inadequate	adequate
	1 2 3 4 5 6 7 8 9	
	useless	helpful
	1 2 3 4 5 6 7 8 9	
1.3.1 Placement of help icons	confusing	clear
	1 2 3 4 5 6 7 8 9	
1.3.2 Accessing help using icons	difficult	easy
	1 2 3 4 5 6 7 8 9	
1.3.3 Finding specific information using the help	difficult	easy
	1 2 3 4 5 6 7 8 9	

PART 2: Wizards

1.1 Wizards were	useless	helpful
	1 2 3 4 5 6 7 8 9	
1.1.1 Accessing the wizards was	difficult	easy
	1 2 3 4 5 6 7 8 9	
1.2 Working through the wizards was	difficult	easy
	1 2 3 4 5 6 7 8 9	

1.2.1	Wizards are meaningfully structured	never always 1 2 3 4 5 6 7 8 9
1.3	Wizard content was	useless helpful 1 2 3 4 5 6 7 8 9
1.3.1	Information for specific aspects of the system was complete and informative	never always 1 2 3 4 5 6 7 8 9
1.3.2	Information was concise and to the point	never always 1 2 3 4 5 6 7 8 9
1.4	Tasks can be completed	with difficulty easily 1 2 3 4 5 6 7 8 9
1.4.1	Instructions given for completing tasks	confusing clear 1 2 3 4 5 6 7 8 9
1.5	Learning to operate parts of the system using the wizard was	difficult easy 1 2 3 4 5 6 7 8 9

Appendix E5: Validation of Questionnaires from Final Evaluation

We conducted reliability and validity tests on all of the questionnaires which we constructed for our final evaluation (Chapter 9). During the pilot study (see Section 9.6.1) we conducted basic reliability and validity tests of the System and QUIS 7 questionnaires. All of the questionnaires were retested with the full data after the experiments, to confirm the pilot study results. We only validated the System and QUIS questionnaires during the pilot study as our sample size was only eight, evenly spread across the four experimental groups. This number was too small to achieve usable results; however, QUIS has been validated previously (e.g., Chin et al. 1988), and we could compare the System questionnaire against it for increased information about validity. Therefore, we tested and refined the System questionnaire in the pilot study, and then validated it, the Representation and the Scaffolding questionnaires after the experiment had been conducted to test whether the questionnaires were valid and therefore usable in the analyses.

Initial Validation of QUIS 7 and the System Questionnaire

We tested the QUIS questionnaire for reliability using Cronbach's alpha. Cronbach's alpha is a measure of the internal consistency or correlation of the items on a scale. It provides a value between 0 and 1. If the value is high it can be interpreted as meaning that the items on the scale measure one construct and are less likely to be measuring noise. There is no set threshold value for Cronbach's alpha, but 0.8 is generally accepted as a minimum for a reliable scale (Anastasi 1996, Cronbach 1990). Although QUIS is already validated, we wanted a comparison with the System questionnaire using our eight participants. This sample size is much too small to determine the reliability of a scale, but we wanted an initial estimate so that we could remove any obviously problematic items from the scale before using it. With eight participants, the QUIS Cronbach's alpha was 0.87 which is acceptable, especially with such a small sample size.

We tested our System questionnaire and gained a Cronbach's alpha of 0.74. When we examined the item-total correlations between all items, we found three problematic items. Item-total correlations measure the extent to which each item measures the same construct as all of the others; an item-total correlation of less than 0.2 indicates a problematic question, and a value of more than 0.4 indicates a good question. The problematic items were a question about using the Fly Mode in the 3D world (correlation = 0.15), and two questions about using multiple windows in VRBridge (correlations = -0.11 and -0.05). We removed these items from the questionnaire, which resulted in a revised scale of 18 items (from an original 21) and a Cronbach's alpha of 0.79. This value was acceptable for the small sample size.

We also tested the concurrent and construct validity of our System questionnaire by calculating how well it correlated with the established QUIS questionnaire. We attained a significant correlation coefficient of 0.78, which confirmed that our scale was concurrently valid with QUIS.

Full Validation

Once we had conducted the full study, we could validate our questionnaires using our full sample of 146 participants. Table E1 shows the results for all three questionnaires that we created. As can be seen from the table, the Cronbach's alpha for all of the questionnaires was high enough that they could be reliably used to analyse our participants' experiences, although that for the System questionnaire is marginal. The average inter-item correlation of a scale indicates how much its items measure the same unidimensional construct on average; for a multi-dimensional scale like QUIS the average inter-item correlation is not likely to be high, and this also reflected in the value for the System questionnaire. Cronbach's alpha is more robust in dealing with multi-dimensional constructs.

Questionnaire	Valid N	Average Inter-Item Correlation	Cronbach's Alpha
QUIS 7	146	0.26	0.95
System	146	0.30	0.87
Representation	73	0.45	0.92
Scaffolding	73	0.65	0.97

Table E1: Reliability and validity data for four questionnaires used in the study, showing the number of participants, average inter-item correlation and Cronbach's alpha.

Because we wanted to break the questionnaires up and measure results from sections dealing with specific parts of VRBridge, we also conducted the above analyses for individual sections. We only did this for the three questionnaires that we created ourselves, as QUIS has already been shown to be robust in this way. The average inter-item correlations and Cronbach's alpha for sections of the three questionnaires are shown in Table E2.

Questionnaire	**Section**	**Average Inter-Item Correlation**	**Cronbach's Alpha**
System	Triggersets	0.64	0.9
	Timelines	0.79	0.94
Representation	Floorplan	0.68	0.93

Questionnaire	Section	Average Inter-Item Correlation	Cronbach's Alpha
	Sequence Diagram	0.7	0.94
	Run Mode	0.75	0.9
Scaffolding	Tooltips	0.93	0.96
	Context Sensitive Hints	0.75	0.93
	Wizards	0.86	0.98

Table E2: Reliability and validity data for sub-sections of three questionnaires used in the study, showing average inter-item correlation and Cronbach's alpha.

As can be seen from Table E2, the average inter-item correlation of all the sub-sections increased, indicating that they were measuring more uni-dimensional constructs than the entire questionnaires. The Cronbach's alpha increased for the System questionnaire, indicating that our measurements of the sub-sections were more reliable than that for the whole questionnaire (i.e. the questionnaire items not included in the selected sub-sections added some noise to the measurement taken by the questionnaire as a whole). As the System questionnaire was the most problematic one (see Table E1), this result is pleasing and allows us to specifically evaluate participants' opinions about Triggersets and Timelines, as opposed to generalised options about the whole system. The values for the Representation and Scaffolding questionnaire sub-sections were high enough that we could be sure of their reliability.

We also retested the correlation between our System questionnaire and QUIS, attaining a correlation coefficient of 0.84 with the increased sample size.

Appendix F: Booklet provided to participants of final VRBridge evaluation

VRBridge Booklet

This booklet is a reference guide for VRBridge and how to use it. Below is a description of virtual environments and how actions work. The rest of the booklet provides some general tips about the system. All of this information will be covered during the tutorial, so don't be concerned if some of it confuses you as it will be explained later. You can refer to the booklet while you are working, to remind you about how to use VRBridge.

First, a definition for **a virtual environment (VE)**: a 3D environment which contains objects, and which a player can move around and interact with. Objects can be stationary e.g. chairs or moving, e.g. simulated people. When you play with this environment on a computer, you are the player of the environment. You can move around the environment and look around, using the keyboard and mouse. In VRBridge, everything in the VE is called an Object. The environment is an Object called the World, and the player is also an object. All other objects have a body which is a model that you can see in the world and for which you can create actions. The most important thing about virtual reality that distinguishes it from films and other media is that the player (or person experiencing it) can move around the world and participate in it.

Anything dynamic that happens in the environment must be created by you. Generally, when you design a VE, the world and objects also have to be created, but you will not have to do this. So, all of the content that you will need is already in the environment. All the objects have animations, e.g. walk, jump, etc, and sounds already attached to them.

To create actions, you use **Triggersets**. Actions are event-based, which means that anything that happens has to follow an event (i.e. something else happening). Triggersets can be activated by the player moving around and using the keyboard, or as a consequence of previously triggered actions.

A triggerset consists of one trigger (a triggering event), one or more conditions that are required for it to go off, and one or more resulting actions.

A trigger is an instant event that happens, like a sound ending or beginning. A trigger can be one of the following kinds:

- Proximity – 2 objects enter or leave a certain distance of each other
- Collision – 2 objects start or stop touching each other
- Visibility – 2 objects start or stop seeing each other
- Location – an object enters or leaves a location (a location is an area of the virtual environment that has been defined)
- User Input – the player presses down or releases a key on the keyboard
- ObjectEvent – an object starts or stops an event. The events can be of type World start (only applicable to the World object), playing a sound, performing an animation or changing a property (these can be object size and whether it is visible or invisible)

A condition refers to the state that the environment is in when the trigger happens. Conditions are of the same type as the triggers above, except that they are not instant.

- Proximity – 2 objects are within or outside a certain distance of each other
- Collision – 2 objects are touching each other or not touching each other
- Visibility – 2 objects can see each other or cannot see each other
- Location – an object is in a location or not in a location
- User Input – the player is holding down or not holding down a key on the keyboard
- ObjectEvent – an object is performing an event, or not performing an event. For conditions, only 3 events are included. These are playing a sound, playing a timeline or performing an animation

An action is something that an object does in the environment. These include:

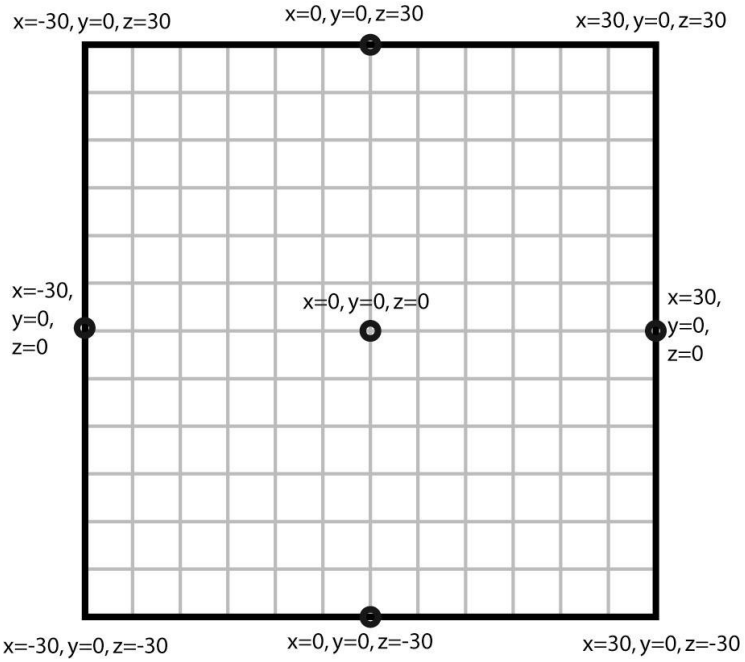
- Animation – an object starts an animation
- Sound – an object starts a sound
- Change Property – an object changes its size or visibility
- Rotate – an object turns around an axis (x, y, or z) for a specified time period, or turns to face another object.
- Move – an object moves a specified distance in a specified time period. The direction can be forwards, backwards, left, right, up or down, or moving towards another object.
- Timeline – an object starts a timeline

So, if the conditions specified in the triggerset are all true (i.e. all happening) when the event specified by the trigger happens, then the actions specified in the triggerset will also happen, possibly setting off other triggersets. This is how the actions progress in the environment.

3D Coordinates

The 3D worlds used in VRBridge can be seen as square boxes in which all the objects are placed. The height of the box is called the y axis, the width of the box is called the x axis and the length of the box is called the z axis. All 3D worlds are structured by these axes. Any point in the box has an x,y,z coordinate, which exactly positions it in the 3D space of the box. The size of our 3D worlds is 60 metres by 60 metres by 60 metres. The spot on the middle of the floor of the box is at x = 0, y = 0, z = 0. This means that the ground is at y = 0, so the height of an object is in y units. It also means that x and z coordinates run from -30 metres to +30 metres. So, an object standing at x = -30, y = 0, z = -30 is standing in one of the four corners of the box,

as is an object standing at x = 30, y = 0, z = 30. The 3D world uses a single 3D coordinate to position an object. The x and z coordinates are shown on the map of the 3D world below.



Selecting Anything in VRBridge

- To select anything in VRBridge, left click on it with the mouse. If there are menus attached to it, they will pop up and then you can select options from them.
- When you have created triggersets, you can left click on the triggers, actions and conditions to view their details and change them.

Creating Triggersets

- You can create triggersets from the InterPlay Screen. The process uses dialog boxes which explain what you have to fill in. Click on the screen where it says “Click to create new trigger”. A dialog box will open so that you can choose the type of trigger to create. When you choose one, a dialog box will open where you select the options to create the trigger. For example, for a collision trigger, you must select the two objects that must collide and whether the trigger goes off when they touch each other or stop touching each other. Remember that you first have to make the objects move in a different triggerset, using the actions, so that they can collide and set off your collision trigger.
- Once you have created a trigger, you can add actions and conditions to it. These are created in the same way as the trigger, except you click on the condition and action spaces.

The 3D Window

- You can open the 3D window immediately from the VRBridge menu, to see what the world that you will be working with looks like.
- Once you have created some triggersets, you can view them immediately in the 3D window. For example, if you create a triggerset where the player presses a key on the keyboard (trigger) and an object

moves and does an animation (actions), you can open the 3D world, press the key that you selected for the trigger, and watch the object do its actions.

- When you open the 3D window, the mouse is used to look around it. If you want to change to another screen, or select any options on the window, press Tab to get the mouse back. Click on the window to use the mouse inside it again.
- You can view the 3D window in Walk or Fly mode by selecting one from the bottom. In fly mode, you can fly up over the VE and in walk mode you move around like the other objects.
- You can pause the running of the 3D window by selecting Pause from the bottom, and play it again by pressing the same button.

Timelines

- Timelines are provided for predictable sequences of actions that will happen together. These are attached to objects, and can be started like any other action. They are also added to the triggers and conditions. Actions can be triggered by a timeline starting or ending, and a timeline happening or not happening can be specified as a condition.
- You can create timelines easily using the system. Just open a new timeline from the Timelines menu item on the InterPlay Screen and start adding actions using the Add button.
- Once you have created actions, you can move them on the timeline. Hover with the mouse over the upper left hand corner of the action box. When the cursor changes to two sideways arrows, you will be able to drag the action.
- To make a timeline play in the 3D world once you have created it, you must attach it to a triggerset as an action (Start Timeline).

Object Panel

- The object panel on the InterPlay screen contains details about all the objects. Click on the Select Object button to change between objects.
- The tabs at the bottom of the screen contain different kinds of information about objects.
- The **Basic** tab contains the name and model name of the object. You can preview the object here by selecting the View Objects button. A window will open with the object in it. Use the forward and back arrow keys to go towards the object and away. Use the left and right arrow keys to swing your view around so you can view different sides of the model.
- The **Physical Attributes** tab contains the initial details of the object: its position (in x, y and z), facing (also x, y and z), size, whether it is visible and whether it has gravity and collision detection. All of these details can be changed and then checked in the 3D world. You can also change the initial position and facing of the player (your point of view through the screen) from here.
- The **Animation** and **Sound** tabs provide a list of all animations and sounds belonging to the object. You can preview them here.
- The **Timelines** tab provides a list of all timelines connected to the object. From here, you can open them or delete them, or create new ones.
- The **User Interface** tab contains the details of how you will move around the world. The default keys are w for forward, s for back, a for left and d for right. You can also change the player's eye-level from here to make yourself taller or shorter.

Waypoints

- Waypoints are 2D points in space (so only the x and z coordinates apply), which you can create to make it easier for objects to move around.
- Waypoints can be used in triggers for proximity and collision. They can also be used in actions, for an object to move or turn towards.

- Create a waypoint by selecting Waypoints from the menu on the InterPlay screen. You must enter the coordinates for the waypoint.
- Waypoints are invisible in the 3D world. You can make them visible by selecting the View Waypoints option on the 3D world menu.

Locations

- Locations are 2D rectangles in space (so only the x and z coordinates apply), which you can create to make it easier to use the space in a VE.
- There is a location trigger type and a location condition type, which can be set off by an object entering or leaving a location.
- Create a location by selecting Locations from the menu on the InterPlay screen. You must enter the coordinates of the upper left and lower right corners of the location. Use the map provided to work out what the coordinates should be.
- Locations are invisible in the 3D world. You can make them visible by selecting the View Locations option on the 3D world menu.

Appendix G: Tables of descriptive statistics for final evaluation

Table G1 displays the means and standard deviations of the task scores for each group (out of a possible score of 40 for each task). The standard deviations for this and our other dependent variables are high. This was expected as the evaluation required creative work by participants, which is subject to large individual differences even without the experimental manipulations.

Level	**N**	**Task 1 (40)**		**Task 2 (40)**		**Task 3 (40)**	
		Mean	**Std.Dev**	**Mean**	**Std.Dev**	**Mean**	**Std.Dev**
Total	124	28.18	9.64	19.58	9.06	23.56	7.63
Control	31	24.13	10.09	14.97	8.82	19.61	8.09
Scaffolding	31	29.42	9.38	19.74	9.57	24.77	7.62
Representation	31	30.19	9.84	22.84	8.14	25.03	7.01
Combination	31	28.97	8.41	20.77	8.16	24.84	6.69

Table G1.: Descriptive statistics of Task Scores for all participants and per group.

Table G2 displays the means and standard deviations for the XML Explore variables. The standard deviations for the *Features* component are very large, even allowing for creative differences. This is largely due to the fact that many participants did not use the features at all, and therefore scored zero, skewing the distribution of this variable.

Variable	**Level**	**N**	**Task 1**		**Task 2**		**Task 3**	
			Mean	**Std.Dev**	**Mean**	**Std.Dev**	**Mean**	**Std.Dev**
XML Explore (60)	Total	124	24.83	8.34	27.48	9.87	32.10	9.17
	Control	31	23.90	8.21	24.29	10.12	29.55	10.93
	Scaffolding	31	26.45	7.56	30.16	9.51	33.90	6.81
	Representation	31	23.94	8.93	29.61	9.58	34.61	8.78
	Combination	31	25.03	8.73	25.84	9.46	30.32	9.05
Variable Sequence (20)	Total	124	9.01	3.14	9.55	3.42	11.60	3.41
	Control	31	8.10	2.64	7.71	3.49	10.00	3.72
	Scaffolding	31	9.26	2.99	9.74	3.31	11.48	3.14
	Representation	31	9.00	3.64	10.45	3.35	12.81	3.21
	Combination	31	9.68	3.12	10.29	2.96	12.13	3.03
	Total	124	4.23	2.48	4.44	2.98	5.88	2.94

Variable	Level	N	Task 1		Task 2		Task 3	
			Mean	Std.Dev	Mean	Std.Dev	Mean	Std.Dev
Inevitable Sequence (10)	Total	124	4.23	2.48	4.44	2.98	5.88	2.94
	Control	31	4.42	2.47	4.61	2.91	5.71	2.85
	Scaffolding	31	4.77	2.64	5.45	3.20	6.68	2.73
	Representation	31	3.94	2.31	3.94	2.85	6.16	3.15
	Combination	31	3.81	2.48	3.74	2.79	4.97	2.87
Creativity (15)	Total	124	5.82	3.45	6.95	3.98	8.70	3.37
	Control	31	4.71	3.36	5.45	4.10	7.68	3.94
	Scaffolding	31	6.42	3.11	7.87	4.02	9.61	3.06
	Representation	31	5.84	3.46	8.23	3.73	9.48	3.30
	Combination	31	6.32	3.75	6.26	3.56	8.03	2.80
Features (5)	Total	124	0.73	1.19	1.73	1.68	1.06	1.58
	Control	31	0.23	0.49	0.74	1.32	0.39	1.17
	Scaffolding	31	0.48	1.06	1.68	1.64	0.55	0.99
	Representation	31	0.84	1.16	2.29	1.83	1.81	1.85
	Combination	31	1.39	1.52	2.19	1.51	1.48	1.75
Player Freedom (10)	Total	124	5.50	1.48	5.78	1.55	5.73	2.65
	Control	31	5.32	1.85	4.97	1.92	4.71	2.30
	Scaffolding	31	5.32	1.35	6.00	1.06	6.19	2.40
	Representation	31	5.55	1.29	5.84	1.57	6.29	2.80
	Combination	31	5.81	1.40	6.32	1.25	5.74	2.90

Table G2: Descriptive statistics for XML Explore total and component scores for total participants and group.

Table G3 presents descriptive statistics of the logging variables collected for all groups. The table organises variables into those used to produce output and those used to gather information. Most of the variables have a large dispersion, with large standard deviations, high maximums and minimums of zero. These large within-group differences are expected for creative work; however recurring similarities in differences in variables between groups show that there are group-based patterns of usage which we can extract.

Variable	Group	Session 1				Session 2			
		Mean	Std.Dev.	Min	Max	Mean	Std.Dev.	Min	Max
VARIABLES FOR PRODUCING OUTPUT									
	Control	22.42	11.99	2	37	17.1	11.40	1	31

Variable	Group	Session 1				Session 2			
		Mean	Std.Dev.	Min	Max	Mean	Std.Dev.	Min	Max
Trig Auth	Control	22.42	11.99	2	37	17.1	11.40	1	31
	Scaffolding	27.94	16.87	4	74	23.68	12.69	2	40
	Representation	32.74	19.11	4	64	25.16	14.32	6	69
	Combination	30.55	16.96	2	59	23.35	15.51	1	56
Timeline Create	Control	6.74	7.43	0	16	6.58	8.57	0	25
	Scaffolding	7.58	8.23	0	29	7.58	7.62	0	16
	Representation	4.13	5.05	0	10	5.23	5.55	0	11
	Combination	4.03	5.99	0	13	4.94	7.29	0	17
Timeline Auth	Control	15.9	24.27	0	88	14.68	18.91	0	52
	Scaffolding	16.81	24.45	0	99	16.68	19.28	0	73
	Representation	8.52	12.81	0	27	12.58	21.16	0	67
	Combination	10.94	18.45	0	49	14.68	26.74	0	83
Object Panel Manip	Control	2.13	6.59	0	15	0.42	2.33	0	8
	Scaffolding	5.13	10.49	0	32	5.16	19.74	0	56
	Representation	9.03	9.17	0	28	3.81	5.68	0	16
	Combination	7.61	9.96	0	25	6.23	15.38	0	32
Waypt Create	Control	0.9	2.21	0	9	0.1	1.33	0	3
	Scaffolding	1.1	1.64	0	6	0.13	2.43	0	4
	Representation	1.87	2.88	0	11	0.61	1.52	0	7
	Combination	2.48	3.04	0	12	1.03	1.77	0	8
Location Create	Control	0.19	1.48	0	3	0.19	0.65	0	3
	Scaffolding	0.19	1.52	0	3	0.13	0.34	0	2
	Representation	0.94	2.53	0	6	0.77	1.43	0	5
	Combination	0.81	2.58	0	6	0.84	1.53	0	5
VARIABLES FOR GATHERING INFORMATION									
Trig Manip	Control	3.77	4.60	0	15	3.58	4.46	0	20
	Scaffolding	2.94	3.71	0	14	3.39	3.69	0	14
	Representation	4.10	4.65	0	16	4.32	3.90	0	15
	Combination	3.10	3.75	0	15	4.16	5.58	0	30
Pause 3D World	Control	1.1	1.04	0	4	0.39	0.56	0	2
	Scaffolding	2.35	2.44	0	9	1.1	1.78	0	8
	Representation	2.74	3.73	0	15	1.32	2.73	0	10
	Combination	2.03	2.65	0	14	1.29	3.47	0	17

Variable	Group	Session 1				Session 2			
		Mean	Std.Dev.	Min	Max	Mean	Std.Dev.	Min	Max
Loc/Way 3D View	Control	0.58	1.34	0	5	0.26	1.26	0	7
	Scaffolding	1.19	2.48	0	11	0.35	1.80	0	10
	Representation	0.45	1.06	0	4	0.03	0.18	0	3
	Combination	0.16	0.58	0	3	0.06	0.36	0	4
Sound Preview	Control	4.61	7.38	0	33	6.94	12.93	0	54
	Scaffolding	6.45	10.07	0	40	8.45	14.64	0	55
	Representation	3.35	5.91	0	24	5.1	6.94	0	26
	Combination	4.84	8.46	0	33	8	11.21	0	41
Anim Preview	Control	5.16	8.75	0	31	7.77	20.00	0	73
	Scaffolding	8.1	17.06	0	54	6.68	18.05	0	67
	Representation	4.13	5.81	0	20	4.9	8.62	0	33
	Combination	3.45	5.37	0	19	3.84	10.47	0	41

Table G3: Descriptive statistics of general computer logging variables, organised by session and group.

Table G4 presents descriptive statistics for the logging variables collected for representations. In general, the dispersion of these variables is not large, although differences in viewing times and their standard deviations are very large. All variables except the Floorplan Open (Representation group) have minimums of zero.

Variable	Group	Session 1				Session 2			
		Mean	Std.Dev.	Min	Max	Mean	Std.Dev.	Min	Max
Open Sequence	Representation	3.13	1.91	0	7	1.71	1.99	0	9
	Combination	3.39	2.67	0	10	1.16	1.13	0	5
Sequence Manip	Representation	3.74	1.69	0	17	2.58	1.56	0	16
	Combination	3.23	1.64	0	14	0.90	0.76	0	9
View Sequence	Representation	5.33	5.52	0	18.40	2.36	3.05	0	15.45
	Combination	10.13	10.41	0	31.45	5.31	11.46	0	33.25
Open Floorplan	Representation	4.45	2.69	1	12	2.94	2.19	0	9
	Combination	4.87	3.40	0	15	3.35	3.63	0	15
Floorplan Manip	Representation	13.29	21.2	0	82	5.68	5.56	0	20
	Combination	10.71	9.49	0	33	8.48	12.13	0	41
View Floorplan	Representation	11.56	13.21	0	56.52	15.07	19.04	0	41.24
	Combination	14.46	16.15	0	51.15	18.24	23.31	0	44.50

Variable	Group	Session 1				Session 2			
		Mean	Std.Dev.	Min	Max	Mean	Std.Dev.	Min	Max
Start Run Mode	Representation	3.10	3.96	0	9	1.68	3.03	0	12
	Combination	2.97	3.86	0	9	0.56	2.25	0	4
View Run Mode	Representation	5.00	4.21	0	19.05	2.38	3.40	0	9.58
	Combination	2.27	2.32	0	8.39	4.25	3.20	0	11.03

Table G4: Descriptive statistics computer logging variables collected for additional representations, organised by session and group.

Table G5 presents descriptive statistics of the logging variables collected for scaffolding. All of the variables have a large dispersion, with large standard deviations compared to their means, relatively high maximums and minimums of zero (except 3D Window Context Hints and therefore Total Hints for the Scaffolding group). For Tooltip usage, we logged whether participants turned them off at the main menu. Since only two participants did this, we did not include a measure of Tooltip usage.

Variable	Group	Session 1				Session 2			
		Mean	Std.Dev	Min	Max	Mean	Std.Dev	Min	Max
Rotation Wizard	Scaffolding	1.03	2.58	0	6	1.55	2.20	0	9
	Combination	0.26	1.25	0	3	0.12	1.32	0	2
Movement Wizard	Scaffolding	2.58	2.21	0	12	3.97	4.63	0	18
	Combination	0.52	1.43	0	4	1.16	3.64	0	8
Context Hints Total	Scaffolding	2.57	2.23	1	16	1.57	3.57	0	16
	Combination	2.56	2.32	0	23	1.55	4.02	0	25
Context Hints 3D Window	Scaffolding	1.20	1.18	1	5	0.12	1.94	0	4
	Combination	0.40	2.03	0	4	0.02	2.07	0	3
Context Hints InterPlay	Scaffolding	0.97	2.30	0	6	0.65	2.25	0	4
	Combination	0.62	1.18	0	4	0.65	3.36	0	5
Context Hints Main Menu	Scaffolding	0.40	3.18	0	5	0.80	3.36	0	8
	Combination	0.28	2.97	0	4	0.16	2.09	0	5
Sequence Hints	Combination	1.16	2.43	0	9	0.52	1.18	0	6
Floorplan Hints	Combination	0.10	0.98	0	2	0.20	0.98	0	3

Table G5: Descriptive statistics of scaffolding computer logging variables, organised by session and group.

The means show that participants did not use the scaffolding very often. However, the Tooltips were always present on the interface. Therefore, all participants who received scaffolding were exposed to the Tooltips constantly. As mentioned above, the log files recorded when participants turned the Tooltips off, but only two participants performed this action during Visit 2, both from the Combination group.

Table G6 displays descriptive statistics for the QUIS 7 questionnaire variables.

Variable	**Level**	**N**	**Session 1**		**Session 2**	
			Mean	**Std.Dev**	**Mean**	**Std.Dev**
QUIS 7 (531)	Total	124	353.01	51.99	392.24	46.26
	Control	31	331.26	56.57	373.97	50.74
	Scaffolding	31	354.03	53.70	390.06	46.77
	Representation	31	371.58	45.26	406.94	43.89
	Combination	31	355.16	45.84	398.00	38.44
Overall (54)	Total	124	31.82	7.32	40.21	6.65
	Control	31	30.55	6.78	38.39	7.32
	Scaffolding	31	29.68	7.24	40.39	6.85
	Representation	31	34.81	6.70	41.87	5.78
	Combination	31	32.26	7.78	40.19	6.41
Screen (81)	Total	124	55.69	10.30	61.86	9.26
	Control	31	52.52	10.06	57.19	9.83
	Scaffolding	31	54.97	13.22	62.32	10.09
	Representation	31	57.55	8.50	64.97	7.59
	Combination	31	57.74	8.23	62.97	7.85
Terminology (162)	Total	124	109.49	16.83	116.87	16.05
	Control	31	101.90	16.66	110.68	17.73
	Scaffolding	31	112.10	16.47	115.10	17.36
	Representation	31	117.06	16.82	125.06	12.95
	Combination	31	106.90	14.02	116.71	12.68
Learning (117)	Total	124	74.95	17.24	85.38	15.54
	Control	31	68.58	18.28	80.97	15.08
	Scaffolding	31	75.29	19.77	84.29	16.69
	Representation	31	77.52	16.32	85.35	16.26
	Combination	31	78.42	12.89	90.90	12.95

Table G6 Descriptive statistics of QUIS 7 Questionnaire Total and Component scores per group and overall.

Table G7 displays descriptive statistics for the System questionnaire variables.

Variable	Level	N	Session 1		Session 2	
			Mean	Std.Dev	Mean	Std.Dev
System Questionnaire (162)	Total	124	116.69	23.06	131.01	18.93
	Control	31	110.26	23.67	128.74	18.05
	Scaffolding	31	116.19	25.03	128.71	16.33
	Representation	31	121.29	19.97	133.55	21.69
	Combination	31	119.00	22.89	133.03	19.62
Triggersets (45)	Total	124	28.76	9.54	34.69	7.56
	Control	31	26.55	9.60	33.23	7.54
	Scaffolding	31	27.26	10.53	33.68	8.44
	Representation	31	30.84	8.37	35.58	5.52
	Combination	31	30.39	9.24	36.29	8.32
Timelines (36)	Total	124	23.57	10.34	27.48	9.34
	Control	31	24.71	9.85	29.97	6.68
	Scaffolding	31	25.87	10.18	29.16	7.60
	Representation	31	22.84	10.30	27.35	9.36
	Combination	31	20.87	10.80	23.45	11.91

Table G7: Descriptive statistics of System Questionnaire Total and Component scores per group and overall.

Table G8 displays descriptive statistics for the Representations Questionnaire variables.

Variable	Level	N	Session 1		Session 2	
			Mean	Std.Dev	Mean	Std.Dev
Representations Questionnaire (144)	Total	62	101.79	24.23	103.79	23.92
	Representation	31	106.33	22.24	107.93	22.71
	Combination	31	97.39	25.60	99.77	24.73
Floorplan (45)	Total	62	35.18	7.56	37.27	5.93
	Representation	31	36.87	6.35	36.87	6.67
	Combination	31	33.48	8.36	37.68	5.18
Sequence Diagram (45)	Total	62	27.76	10.01	27.32	10.89
	Representation	31	27.61	10.32	28.42	10.55
	Combination	31	27.90	9.86	26.23	11.28

Variable	Level	N	Session 1		Session 2	
			Mean	Std.Dev	Mean	Std.Dev
Run Mode (27)	Total	62	11.65	4.34	11.65	5.14
	Representation	31	12.97	4.03	12.94	4.72
	Combination	31	10.32	4.29	10.35	5.29

Table G8: Descriptive statistics of Representations Questionnaire Total and Component scores per group and overall.

Table G9 displays descriptive statistics for the Scaffolding Questionnaire variables.

Variable	Level	N	Session 1		Session 2	
			Mean	Std.Dev	Mean	Std.Dev
Scaffolding Questionnaire (198)	Total	62	137.75	39.34	146.02	43.15
	Scaffolding	31	131.79	39.79	152.31	38.02
	Combination	31	144.38	38.50	139.00	48.02
Tooltips (18)	Total	62	13.03	3.31	13.18	2.89
	Scaffolding	31	13.48	3.54	13.03	3.49
	Combination	31	12.58	3.06	13.32	2.20
Context Sensitive Hints (45)	Total	62	32.47	8.43	33.58	6.56
	Scaffolding	31	31.32	10.36	32.68	7.29
	Combination	31	33.61	5.86	34.48	5.71
Wizards (108)	Total	62	70.66	16.73	75.45	14.77
	Scaffolding	31	70.29	18.55	75.23	16.47
	Combination	31	71.03	14.99	75.68	13.12

Table G9: Descriptive statistics of Scaffolding Questionnaire Total and Component scores per group and overall.