

A VIRTUAL ENVIRONMENT AUTHORIZING
INTERFACE FOR CONTENT-EXPERT AUTHORS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE,
FACULTY OF SCIENCE
AT THE UNIVERSITY OF CAPE TOWN
IN FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By

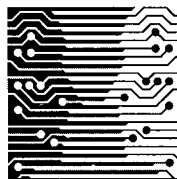
Jakkaphan Tangkuampien

February 2005

Supervised by

Gary Marsden

Edwin H. Blake



The copyright of this thesis vests in the author. No quotation from it or information derived from it is to be published without full acknowledgement of the source. The thesis is to be used for private study or non-commercial research purposes only.

Published by the University of Cape Town (UCT) in terms of the non-exclusive license granted to UCT by the author.

Copyright 2005
by
Jakkaphan Tangkuampien

Abstract

Since the advent of virtual reality (VR), the technology has been exploited in many areas to aid information transfer. In this respect, virtual reality can be regarded as a medium across which authors can communicate with a target group. However, many experts in non-computer-related areas, looking to exploit VR often come unstuck trying to take advantage of this medium. In these cases, one cannot blame these content-expert authors as they have successfully exploited other media prior to VR. On the other hand, the fault can not lie with the medium itself since it has been effectively exploited by other groups of authors. One probable cause could be the authoring tools themselves, or rather their interfaces to be more accurate. A tool's authoring interface is the only access point into the VR medium and one can only assume that the interfaces are not doing their job effectively.

Our study was aimed at investigating authoring interfaces especially from the point of view of content-expert authors. Our approach was to involve such authors who have been able to master existing authoring tool mostly on their own. These authors were in a unique position – having managed to overcome initial difficulties, they have come to understand the inner working of the medium itself.

The study was also well-suited to the appreciative inquiry (AI) methodology – a community-centric methodology that has rarely been applied in the area of computer science. Appreciative inquiry, with its root in action research, encourages a similar spiral-based methodology but with positive approach in all phases. With a group of content-expert VR authors, we applied a cycle of AI, resulting first in a list of interface issues that required some attention as well as some idea of how they can be resolved. The second phase of AI involved working closely with the authors to come up with resolution strategies to each of these issues. These solutions were then assessed for the level at which they have addressed their respective issue by another group of content-expert authors. Finally, an online survey was conducted to extend our results to the wider population of content-expert authors. The survey results confirmed that the interface issues discovered applied to the general population and that the proposed solutions were generally thought to be advantageous to the authoring process. Additionally, these positive results were encouraging since it means that our adaptation of AI was successful.

Acknowledgement

I would like to thank the following people.

- All those who participated in all interview sessions. They gave me more than just their free time and a few free lunches. Without them, there would have been no study.
- Anonymous authors who have filled in my web survey. They took the time to sit through and all my seemingly unnecessary questions and in turn provided me with valuable information.
- Architecture students who took the time to describe and show me their software packages. Thanks must also go to those other architects who assisted me.
- Leanne and Maria for letting me spend time and see what they have created to help teach their little pupils. Thanks also to the little ones who have given me wonderful comments and ideas, some of which I could have never come up with myself.
- Murray and his colleagues for sitting up with me till late in more than a few occasions, to help me with my questions, all the way from Scotland. Thanks also for their patient in putting up with all the different communication packages we have tried to use.
- My research colleagues who sat down with me many different times to help, especially Sarah, Cara, Dave, Ilda, Hendranus, Masophia, Rudi, Sabeeha and Zayd. They all took valuable time from their own research in order to help me.
- My friends who have kept me sane through out the entire process.
- Thanks to both my supervisors who have helped to point me in the right direction, even though sometimes I did get lost, but those occasions were probably due to the faults of my own.
- Others, whom I might have forgotten about, have all helped me tremendously.

Table of Content

Abstract.....i

Acknowledgement ii

Table of Content iii

Lists of Tables vi

Lists of Figures vii

Chapter 1: Introduction 1

 1.1 Main Aims 2

 1.2 Research Outline.....3

 1.3 Outline of this Dissertation 3

Chapter 2: Virtual Environments and Virtual Environment Authoring.....5

 2.1 Virtual Environments 6

 2.1.1 Virtual Reality 6

 2.1.2 Virtual Reality as a Medium 7

 2.1.3 Effectiveness of VEs 10

 2.2 VE Authoring and VE Authoring Tools.....12

 2.2.1 Virtual Environment Authoring Methodologies 12

 2.2.2 Virtual Environment Authoring Tools.....14

 2.2.3 Authoring and Display Platforms.....15

 2.3 Summary.....16

Chapter 3: Relevant Research Methodologies 17

 3.1 Action Research 18

 3.1.1 Introduction.....18

 3.1.2 Action Research in Information Systems Research 19

 3.2 Appreciative Inquiry (AI).....20

 3.2.1 The four Ds Cycle of Appreciative Inquiry.....20

 3.2.2 Positive Energy 21

 3.2.3 Underlying assumptions 22

 3.2.4 When to use AI.....23

 3.2.5 What do with Negative Data 23

 3.2.6 Determining the End.....24

 3.2.7 Evaluation.....24

 3.3 Summary.....24

Chapter 4: Methodology.....25

 4.1 Adaptation of Appreciative Inquiry 26

 4.1.1 Why use AI? 26

 4.1.2 Detail of Adaptation.....26

 4.2 Identification of authors 28

 4.3 Discovery Interviews 30

 4.4 Consolidation Interviews 31

 4.5 Solution Design Cycles 32

 4.6 Evaluation Interviews.....34

 4.7 Verification Survey.....35

4.8 Overall Methodology	37
Chapter 5: Interface Issues.....	39
5.1 Multiple Tools (P1).....	41
5.1.1 Comments.....	42
5.1.2 Origin of Issue	47
5.2 Compile and View Model (P2)	48
5.2.1 Comments.....	48
5.2.2 Analysis.....	53
5.3 Mixed 2D/3D Interfaces (P3).....	54
5.3.1 Comments.....	54
5.3.2 Analysis.....	58
5.4 Multiple Viewpoints Object Placement Interface (P4)	59
5.4.1 Comments.....	59
5.4.2 Analysis.....	62
5.5 Selecting and Editing Objects (P5)	63
5.5.1 Comments.....	63
5.5.2 Analysis.....	66
5.6 Lack of Real World Mappings (P6)	67
5.6.1 Comments.....	67
5.6.2 Analysis.....	72
5.7 Interaction Specification Interface (P7).....	73
5.7.1 Comments.....	73
5.7.2 Analysis.....	78
5.8 Time Manipulation (P8).....	79
5.8.1 Comments.....	79
5.8.2 Analysis.....	82
5.9 Summary.....	82
Chapter 6: Solutions and Resolutions	85
6.1 Solution Overview.....	86
6.2 VE clock & Unified WYSIWYG Interface.....	88
6.2.1 Temporal Manipulation.....	88
6.2.2 Unified WYSIWYG Interface	90
6.2.3 What do VE Clock & Unified WYSIWYG Interface solve?.....	91
6.3 Interaction Subsystem.....	92
6.3.1 Background	92
6.3.2 Authoring Actions.....	93
6.3.3 VE-Based Actions	95
6.3.4 Monitor-based Actions	97
6.3.5 Interaction Manager, Undo and Time Handlers.....	98
6.3.6 Interaction Subsystem Overview	100
6.4 In-Environment Interface	101
6.4.1 Introduction.....	101
6.4.2 3D Cursor.....	102
6.4.3 Object Placement and Selection Interface	103
6.4.4 Revolving Menu	105
6.4.5 In-Environment Console Bar	107
6.4.6 Speech Bubble Interface.....	108
6.5 Object-based Interaction Specification Interface	109
6.5.1 Visual Specification Interface.....	109

6.5.2 Advanced Interface	111
6.5.3 What does this solves?	111
6.6 Objects and Objects Library	111
6.6.1 Objects	111
6.6.2 Objects Library	112
6.6.3 Special Objects: The Avatar and World Objects.....	112
6.6.4 The Avatar Object.....	113
6.6.5 The World Object.....	113
6.7 Resolution Strategies.....	114
6.7.1 Multiple Tools	115
6.7.2 Compile and View Model	116
6.7.3 Mixed 2D/3D Interfaces.....	117
6.7.4 Multiple View-points Object Placement Interface	118
6.7.5 Selecting and Editing Objects	119
6.7.6 Lack of Real World Mappings	120
6.7.7 Interaction Specification Interface.....	121
6.7.8 Time Manipulation.....	122
Chapter 7: Evaluations & Verification.....	123
7.1 Overview	124
7.2 Multiple Tools.....	125
7.3 Compile and View Model	126
7.4 Mixed 2D/3D Interfaces.....	128
7.5 Multiple View-points Object Placement Interface	129
7.6 Selecting and Editing Objects	131
7.7 Lack of Real World Mappings.....	132
7.8 Interaction Specification Interface.....	135
7.9 Time Manipulation.....	138
7.10 Summary.....	140
Chapter 8: Conclusions.....	141
8.1 Virtual Environment Authoring Interface	142
8.1.1 Summary of Issues and Resolution Strategies.....	142
8.1.2 Problems Encountered and Potential Future Works	144
8.2 Appreciative Inquiry Adaptation.....	145
8.2.1 Lessons from Methodological Adaptation.....	145
8.2.2 Suitability of Appreciative Inquiry	147
8.2.3 Future Work for AI Adaptation.....	147
8.3 Meeting the Aims	148
Appendices	149
Appendix A: Using VEBUILDER Booklet.....	150
Appendix B: Survey Invitation E-mails.....	163
Appendix C: Online Verification Survey Questions	165
Bibliography.....	200

Lists of Tables

Table 1: Summary of the top eight issues related to using virtual environment authoring tools, comments on each by a wide group of authors, and analysis of possible reasons why each is present in a number of virtual authoring tools used by these authors.....83

Table 2: Summary of the results of the verification survey 140

Lists of Figures

Figure 1:	Virtual reality as a communication medium - a product of reality-based and non-reality based media.	9
Figure 2:	Authoring for the VR medium: While packaged information in the VE may be accurately transferred from the content author to the consumer, there is a chance of a mistranslation from the author's idea into VE.	11
Figure 3:	Kaur's Stages in creating a VE	13
Figure 4:	Existing types of tools and how they fit into Kaur's stages of VE authoring.....	15
Figure 5:	Task Cycles in Action Research.....	18
Figure 6:	Tasks cycle in action research in the field information systems	19
Figure 7:	Tasks cycle in appreciative inquiry.....	21
Figure 8:	Methodology employed in our research – an adapted version of appreciative inquiry. The discovery and consolidation interviews formed AI discovery and dream phases. Part of the consolidation interviews also contributed as the starting point for the design process. The evaluation process and verification process acts as an endpoint to the design process and also as a kick off point to the delivery process by introducing our solution to the general population group. In total three different groups of VE authors took part; refer to 4.2 Identification of authors for more detail.	28
Figure 9:	Summary of discovery and consolidation phases.....	32
Figure 10:	Summary of Solution Development Cycles.....	34
Figure 11:	Summary of Evaluation Process.....	36
Figure 12:	Overall Methodology	38
Figure 13:	VE Authoring with multiple tools or interfaces.....	41
Figure 14:	Question 5.2.1.4 from the survey which enquired authors about confusion in using multiple tools. (NA indicates users not affected by the issue.).....	42
Figure 15:	Question 5.2.1.1 from the survey which enquired authors on incompatibility aspect of using multiple tools. (NA indicates users not affected by the issue.).....	42
Figure 16:	Question 5.2.1.2 from the survey which enquired authors on the non-integration aspect of using multiple tools. (NA indicates users not affected by the issue.).....	43
Figure 17:	Question 5.2.1.3 from the survey which enquired authors on multiple steps mentality of using multiple tools. (NA indicates users not affected by the issue.).....	43
Figure 18:	VE building habits, grouped into those using multiple tools and those that do not. (Results from PS Survey Question 4.2.1.5, cross-referenced with Question 5.1.1.1).....	43
Figure 19:	Question 5.2.1.5 from the survey which enquired authors on amount of saves when using multiple tools. (NA indicates users not affected by the issue.)	44

Figure 20:	Question 5.3.1.2 from the survey which enquired authors on the effect of using multiple tools on their authoring process. (NA indicates users not using multiple tools.)	Error! Bookmark not defined.
Figure 21:	Amount of time authors think it would take them to author a small VE, grouped into those using multiple tools and those that do not. (Results from PS Survey Question 4.2.1.2, cross-referenced with Question 5.1.1.1)	Error! Bookmark not defined.
Figure 22:	Amount of time authors took to master their tool(s), grouped into those using multiple tools and those that do not. (Results from PS Survey Question 4.2.1.1, cross-referenced with Question 5.1.1.1)	46
Figure 23:	Question 5.2.2.1 from the survey which enquired authors on the learning aspect of using multiple tools. (NA indicates users not affected by the issue.)	46
Figure 24:	Question 4.2.2.3 from the survey, which enquired authors on the effect of using multiple tools on the quality of their VEs. (NA indicates users not using multiple tools.)	46
Figure 25:	Question 6.2.1.4 from the survey, which enquired authors on the difference in representation between edit and view modes. (NA indicates users not affected by this issue.)	48
Figure 26:	Question 6.3.1.3 from the survey, which enquired authors on the effect of using compile/view tools on the quality of their VEs. (NA indicates users not using compile/view tools.)	49
Figure 27:	Question 6.2.1.1 from the survey, which enquired authors on the compile/build frequency while using compile-view tools. (NA indicates users not using compile-view tools.)	49
Figure 28:	Question 6.2.1.2 from the survey, which enquired authors on amount of work done in between compile/build steps while using compile-view tools. (NA indicates users not using compile-view tools.)	50
Figure 29:	Amount of time authors took to master their tool(s), grouped into those using multiple tools and those that do not. (Results from PS Survey Question 4.2.1.1, cross-referenced with Question 5.1.1.1)	50
Figure 30:	Question 6.2.1.5 from the survey, which enquired authors about interface differences in their compile-view tools. (NA indicates users not using compile-view tools.)	50
Figure 31:	Question 6.3.1.2 from the survey, which enquired authors on the effect of using compile and view tools on the authoring process (NA means authors not using compile& view tools)	51
Figure 32:	Amount of time authors took to master their tool(s), grouped into those using compile/view tools and those that do not. (Results from PS Survey Question 4.2.1.1, cross-referenced with Question 5.1.1.1)	51
Figure 33:	Question 6.3.1.1 from the survey, which enquired authors on the effect of using compile and view tools on the learning process (NA means authors not using compile& view tools)	52
Figure 34:	Question 6.3.1.3 from the survey, which enquired authors on the effect of using compile& view tools on the quality of their VEs. (NA indicates users not using compile& view tool.)	52
Figure 35:	Question 8.2.2.4 from the survey, which enquired authors on the difference in pointers in 2D and 3D portion of their mixed	

	interface (NA represents users for whom there are no apparent separation between 2D and 3D)	54
Figure 36:	Question 8.2.2.1 from the survey, which enquired authors on mindset change when authoring in 2D and 3D portions of their tools (NA indicates users for whom there are no separation between 2D and 3D-based interfaces)	55
Figure 37:	Question 8.2.2.2 from the survey, which enquired authors about 2D interface clutter on their tools (NA indicates users for whom there is no separation between 2D and 3D-based interfaces).....	55
Figure 38:	Question 8.2.2.3 from the survey, which enquired authors the amount of view area dedicated for the 3D portion of their tools. (NA indicates users for whom there is no separation between 2D and 3D-based interfaces)	55
Figure 39:	Screen captures from GenEdit (left) and Maya (right) in single pane mode. Each shows the relative default sizes of 2D and 3D aspects of the interface.	56
Figure 40:	Question 8.2.2.3 from the survey, which enquired authors about the amount of view area dedicated for the 3D portion of their tools. (NA indicates users for whom there is no separation between 2D and 3D-based interfaces)	56
Figure 41:	Question 8.2.2.5 from the survey, which enquired authors on how they see the difference in 2D/3D portions of the interface. (NA indicates users for whom there is no separation between 2D and 3D-based interfaces)	57
Figure 42:	Question 8.2.1.4 from the survey, which enquired authors on the difficulty of changing physical properties of an object in the virtual environment.	57
Figure 43:	Results from survey Questions 8.3.1.1-8.3.1.3 which enquired the users on the effects on learning, authoring and quality of VE, by the 2D/3D mixture of user interfaces in the tools that they used.	58
Figure 44:	Maya's default interface has four sub-panes, each showing a different viewpoint – the user may change to a single pane mode (see Figure 39).	59
Figure 45:	Question 7.2.2.1 from the survey, which enquired authors on the complexity of placing or selecting objects using the multiple-pane placement interface. (NA indicates authors not using this type of interface)	60
Figure 46:	Question 7.2.2.3 from the survey, which enquired authors on the complexity of placing or selecting objects using the multiple-pane placement interface. (NA indicates authors not using this type of interface)	60
Figure 47:	Question 7.2.2.2 from the survey, which enquired authors on the complexity of placing or selecting objects using the multiple-pane placement interface. (NA indicates authors not using this type of interface)	60
Figure 48:	Difficulty rating for the task of moving an object while editing, grouped into those using multiple pane/viewpoint placement interface and those that do not. (Results from PS Survey Question 4.2.2.2, cross-referenced with Question 7.1.1.1).....	61
Figure 49:	Question 7.2.2.2 from the survey, which enquired authors on the complexity of placing or selecting objects using the multiple-pane	

	placement interface. (NA indicates authors not using this type of interface)	61
Figure 50:	Results from survey Questions 7.3.1.1-7.3.1.3 which enquired the users on the effects on learning, authoring and quality of VE, by the multiple pane placement interface.	62
Figure 51:	Question 9.2.2.3 from the survey, which enquired authors on the level of confusion related to indirect selection. (NA indicates authors not using this type of interface).....	63
Figure 52:	Question 9.2.2.1 from the survey, which enquired authors on the level of difficulty related to indirect selection. (NA indicates authors not using this type of interface).....	64
Figure 53:	Difficulty rating for the task of finding and selecting an object while editing, grouped into those using direct and indirect methods. (Results from PS Survey Question 4.2.2.2, cross-referenced with Question 9.1.1.1).....	64
Figure 54:	Question 9.2.2.4 from the survey, which enquired authors on the level of difficulty in keeping track of current object using indirect selection. (NA indicates authors not using this type of interface).....	64
Figure 55:	Question 9.2.2.2 from the survey, which enquired authors on time taken to find objects using indirect selection. (NA indicates authors not using this type of interface).....	65
Figure 56:	Results from survey Questions 9.3.1.1-9.3.1.3 which enquired the users on the effects on learning, authoring and quality of VE, by the direct and indirect object selection interface in the tools that they used.....	65
Figure 57:	Question 12.2.1.2 from the survey, which enquired authors on how they place objects on the floor. (NA indicates not applicable)	67
Figure 58:	Question 12.2.1.1 from the survey, which enquired authors on how they place objects next to walls or other objects. (NA indicates not applicable)	67
Figure 59:	Question 11.2.1.1 from the survey, which enquired authors on defining objects' behaviour. (NA indicates not applicable)	68
Figure 60:	Question 11.2.1.2 from the survey, which enquired authors on whether they have to multiply define the same behaviours for similar objects (NA indicates not applicable).	68
Figure 61:	Question 13.2.1.1 from the survey, which enquired authors on the how they can add backdrop/skybox and/or weather conditions to their VE. (NA indicates not applicable)	69
Figure 62:	Question 13.2.1.2 from the survey, which enquired authors on the complexity of adding backdrop/skybox and/or weather conditions in their VE. (NA indicates not applicable).....	69
Figure 63:	Question 4.2.2.7 from the survey, which asked authors to give difficulty rating for the task of changing global setting such as location, weather condition or time of day.	69
Figure 64:	Results from survey Questions 11.3.12, 13.3.1.2 and 12.3.1.2 which enquired the users on the effects on authoring process, by a) lack of gravity and collision detection, b) lack of object mapping to real world counterparts and c) time consuming global environment specification of weather condition and time of day.....	70
Figure 65:	Results from survey Questions 11.3.11, 13.3.1.1 and 12.3.1.1 which enquired the users on the effects on learning process, by a) lack of	

	gravity and collision detection, b) lack of object mapping to real world counterparts and c) time consuming global environment specification of weather condition and time of day.....	70
Figure 66:	Results from survey Questions 11.3.13, 13.3.1.3 and 12.3.1.3 which enquired the users on the effects on the overall quality of virtual environments by a) lack of gravity and collision detection, b) lack of object mapping to real world counterparts and c) time consuming global environment specification of weather condition and time of day.	71
Figure 67:	Question 14.2.1.1 from the survey, which enquired authors on whether they had to specifically learn how to program in order to create a VE.	73
Figure 68:	Question 14.2.1.2 from the survey, which enquired authors on the difficulty of adding interactions or behaviours by programming or scripting.....	74
Figure 69:	Question 14.2.1.3 from the survey, which enquired authors on whether they avoid adding complex interactions because of difficulty with the specification.	74
Figure 70:	Question 14.2.1.6 from the survey, which enquired authors on whether they do the programming/scripting in a separate interface.....	74
Figure 71:	Question 11.2.1.2 from the survey, which enquired authors on problems stemming from the compilation aspects of their interaction specification.	75
Figure 72:	Question 14.2.1.7 from the survey, which enquired authors about the visual linkage between objects which are in an interaction.....	75
Figure 73:	Question 4.2.2.6 from the survey, which asked authors to rate how difficult it is to find objects that are in an interaction.	76
Figure 74:	Question 4.2.2.5 from the survey, which asked authors to give difficulty rating for the task of adding interactions between two or more objects.	76
Figure 75:	Question 14.2.1.4 from the survey, which enquired authors on time taken to create a simple interaction via programming. (NA indicates authors not using this type of interface).....	77
Figure 76:	Question 14.2.1.5 from the survey, which enquired authors on whether they spend a lot of time repeatedly programming similar interactions (NA indicates authors not using this type of interface)	77
Figure 77:	Results from survey Questions 14.3.1.1-14.3.1.3 which enquired the users on the effects on learning, authoring and quality of VE, using programming or scripting to create interactions.....	77
Figure 78:	Question 15.2.1.2 from the survey, which enquired authors on the time spend waiting for a sequence to take place because there is no fast forward functionality.	79
Figure 79:	Question 15.2.1.1 from the survey, which enquired authors on the difficulty in checking that simultaneous actions occur correctly.....	79
Figure 80:	Question 4.2.2.8 from the survey, which asked authors to rate how difficult it is to create a time-based sequence of events.....	80
Figure 81:	Question 15.2.1.3 from the survey, which enquired the authors on whether they would like the pause, fast forward and rewind functionalities.....	80

Figure 83:	Results from survey Questions 15.3.1.1-15.3.1.3 which enquired the users on the effects on learning, authoring and quality of VE, by the lack of time manipulation functionality.....	81
Figure 84:	Screenshots of the temporal manipulation control, from first person view (top) and from bird's eye view (bottom)	86
Figure 85:	View from the top.....	86
Figure 86:	Various aspects of the in-environment interface.....	86
Figure 87:	Interaction specification interface: Creating interaction from first person view (top-left), from bird's eye view (top-right); editing an object's attributes (middle) and detailed interface for specifying interaction (bottom).....	87
Figure 88:	Representation of avatars from different viewpoints.....	87
Figure 89:	VCR-like control interface for (from top) Media Player, WinAmp and Real Player.....	88
Figure 90:	Screen capture from VEBUILDER II, showing the WYSIWYG interface and the time manipulation console.....	89
Figure 91:	A possible task breakdown for a VE authoring process. The two main tasks of environment building and interaction specification are broken down into smaller subtasks (not all are shown). Some tasks are broken further to even smaller steps or authoring actions.....	94
Figure 93:	Visual overview of the interaction subsystem.....	100
Figure 94:	Screen Capture from VEBUILDER I, showing the 3D pointer and its reach. The pointer is manipulated using the mouse while the avatar's movement is manipulated using the keyboard.....	102
Figure 95:	Screen captures from VEBUILDER I, showing how new object can be placed inside a virtual environment. When placing objects, the 3D pointer points downwards to indicate the target position. A rotating red cross also appears.	104
Figure 96:	Screen capture form VEBUILDER I, showing the 3D revolving menu interface.....	106
Figure 97:	Balloon help interface from Mac OS (top) and Microsoft's version from Windows XP (bottom)	108
Figure 98:	Screen capture from VEBUILDER III, showing the speech bubble interface. This interface can be used to manipulate objects or create an interaction.	108
Figure 99:	Screen Capture from VEBUILDER III, showing the author in the middle of specifying an interaction. The wall has already been selected as a watcher. The yellow line indicates an interaction link. The author must now select and configure another object to complete the specification. First person view (top) and top view (bottom).....	110
Figure 100:	Screen capture from VEBUILDER I showing the avatar representation in first person view (left) and top view (right).....	112
Figure 101:	Survey result for Question set 5.4 which presented affected authors with a brief description of the resolution concepts for P1: Multiple Tools. Question 5.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.....	126
Figure 102:	Survey result for Question set 6.4 which presented affected authors with a brief description of the resolution concepts for P1:	

	Compile and View Model. Question 6.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.	127
Figure 103:	Survey result for Question set 8.4 which presented affected authors with a brief description of the resolution concepts for P3: Mixed 2D/3D Interface. Question 8.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.	129
Figure 104:	Survey result for Question set 7.4 which presented affected authors with a brief description of the resolution concepts P4: Multiple viewpoints object placement interface. Question 7.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.	130
Figure 105:	Survey result for Question set 9.4 which presented affected authors with a brief description of the resolution concepts P5: Selecting and Editing Objects. Question 9.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.	132
Figure 106:	Survey result for Question set 11.4 which presented affected authors with a brief description of the resolution concepts P6: Lack of real world mappings (Objects' behaviours). Question 11.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.	133
Figure 107:	Survey result for Question set 12.4 which presented affected authors with a brief description of the resolution concepts P6: Lack of real world mappings (Gravity and collision detection). Question 12.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.	134
Figure 108:	Survey result for Question set 13.4 which presented affected authors with a brief description of the resolution concepts P6: Lack of real world mappings (Global Environment). Question 13.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.	135
Figure 109:	Survey result for Question set 14.4 which presented affected authors with a brief description of the resolution concepts P7 Interaction specification interface. Question 14.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.	137
Figure 110:	Survey result for Question set 10.4 which presented affected authors with a brief description of the resolution concepts P7 Interaction specification interface (Interaction link visualization). Question 10.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.	138
Figure 111:	Survey result for Question set 15.4 which presented affected authors with a brief description of the resolution concepts P8 Time Manipulation. Question 15.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.	139

1

Introduction



In 1995, PC Magazine published an article entitled “Virtual Reality - Virtually Here” [68]. This was one of the first virtual reality (VR) articles written for the general public. Merely a few months after low-cost 3D accelerator chips became available for home computers, the hype surrounding VR was already in full force; the message was clear: anyone can create a virtual world on their own desktops. A decade and one eon in the computing world later, is virtual reality really here for everyone? Is everyone empowered enough to successfully exploit this medium?

A few months after the publication of the PC Magazine article, Sherman *et al.* [58] discussed the idea of literacy in virtual reality, both in terms of content receiver and creator. It was clear then that virtual reality needs to develop a lexicon (i.e. a user interface) in which content creators can effectively specify their virtual worlds. This dissertation focuses on whether or not content creators feel that virtual environment authoring tools offer effective lexicons and what can be done to improve the current situation.

We used a user-centric approach to discovering issues with, and subsequently suggest improvements to, currently available tools. The users in our case were virtual environment authors from a wide range of fields (such as education and entertainment). They did have one thing in common though; their desire to use virtual environments in their own application area. We refer to these authors as content-expert authors.

From the start of our research the same group of authors was involved in identifying interface issues as well as finding effective solutions for them. A different group of authors (from similar backgrounds and level of experience as the other group) took part in the evaluating how well each issue was addressed by the proposed solutions. Lastly, an extensive online survey was conducted and authors from wider application areas took part. This survey focused on two issues – firstly to confirm that they are affected by the issues and secondly to evaluate the solution. This was done in order to find out whether our study can be expanded to the more general population of VR authors.

We employed the appreciative inquiry (AI) methodology in our research. Appreciative inquiry is an adaptation of action research with more focus on positive interview techniques to acquire results. These results are then used to make changes to the current version of the system and the cyclic process continues.

1.1 Main Aims

This project’s main aims were to:

1. Adapt the appreciative inquiry research methodology to identify interface issues experienced by the majority of content-expert authors who have looked to apply VR in their own application areas.

2. Propose improvements to the interface and evaluate them for their effectiveness in addressing the respective issue.
3. Evaluate the suitability of appreciative inquiry-based research to computer science.

1.2 Research Outline

Appreciative inquiry has been employed mostly in the corporate world to affect positive change [21]. It has also been used successfully in improving the quality of life in poor or troubled communities [28][21]. It works by focusing primarily on the positive outcomes and assumes that the community in question has the capacity to improve, whether it is in managing a large company or in farming for the village.

Software users also form a community, not unlike one found in a company or a village. They have learnt to use their software effectively and also have knowledge of what they want the software to do. They expect their software to help them do their job but yet this is not always the case. We intend to treat virtual environment authors as a community and apply appreciative inquiry to improve upon their authoring process. To do this, we look to how action research, itself related to appreciative inquiry, was adapted by the information systems community.

1.3 Outline of this Dissertation

Chapters 2 and 3 concern all related background work.

Chapter 2: Virtual Environments and Virtual Environment Authoring

In this chapter, we discuss the concept of virtual reality and how we can think of it as a medium of communication via the creation of virtual environments (VEs). In the second part of the chapter, we also looked at VE authoring and the different methodologies that have been employed.

Chapter 3: Relevant Research Methodologies

We discuss relevant research methodologies in this chapter; starting with action research and how it has been adopted for information systems related areas. We also discuss appreciative inquiry (AI) and its application and related issues. We conclude the chapter with a discussion on the suitability of AI for study in the user interface area.

In chapters 4-7, we present the process of identifying the interface issues with currently available VE authoring tools. During this process, the solutions to these issues are also proposed and these are then incorporated into a set of prototypes and interface mock-ups to be used in the evaluation process.

Chapter 4: Methodology

In this chapter we describe our adaptation of AI methodology as well as outline the steps in which we took in the study.

Chapter 5: Interface Issues

This chapter starts off with the identification of problems affecting the target user groups by mean of interviews. General surveys of authors who have created virtual environments were conducted. These surveys and follow-up interviews with the target user sample group were used to confirm and prioritise the different problems identified previously.

Chapter 6: Solution and Resolutions

The solutions to the interface issues from the previous chapter are proposed. As part of the solution design process, prototypes and interface mock ups incorporating these solutions were presented to the authors and discussed several times. This chapter presents the process that took place and the resultant solutions.

Chapter 7: Evaluation and Verification

We present the result of interviews that were conducted with another set of experts. This group of experts gave independent evaluations of the solutions. We also present the result of a survey that was conducted to verify that the issues and solutions were also applicable to the wider community of VE authors

Chapter 8: Conclusions

In this chapter, we present a summary of the results, lessons learnt, future work and conclusions.

2

Virtual Environments and Virtual Environment Authoring

In this chapter, we present a brief overview of virtual reality (VR) and virtual reality environments (often referred to as virtual environment or just VEs) that can be created within it. We also look at the process of VE authoring, with specific focus on those authors who are expert in their own field, looking to explore VR for their own reasons. We refer to these authors as content-expert authors.



2.1 Virtual Environments

In this section we work towards defining the terms *virtual reality* (VR) and *virtual environments* (VE) – sometimes referred to as *virtual reality environments* (VRE). We proceed in this section by providing a wide variety of definitions for VR.

2.1.1 Virtual Reality

There is no single definition for VR which is agreed upon by all involved. There are a number of definitions that have evolved since the term was first coined. Most of these were based on the simplest definition of VR, namely any computer simulated environment where the user can interact with objects and other users. A definition along this line was the Merriam-Webster entry recorded in 1989, which defined VR simply as

“An artificial environment which is experienced through sensory stimuli (as sights and sounds) provided by a computer and in which one’s actions partially determine what happens in the environment” [79]

This definition implied that even 2D-based multimedia software (e.g., Flash animations) that allows some type of user interactions should be classified as VE. Over time, further refinements to this basic definition usually served to provide a lowest cut-off point to exclude certain type of interactive computer simulated environments from the definition. There are generally two factors which have been used as criteria to provide this cut-off point: technological level of the system and level of experience or feeling provided to the user by the system.

Many definitions of VR require certain levels of technology in the supporting equipment and hardware. One such definition provided by Summit [66] is that virtual reality is a simulation of an environment experienced via a head-mounted display and other wired sensory devices. Some definitions make some requirements as to the nature of the environment. For example, the definition for VR in Encyclopaedia Britannica [22] is given as:

“The use of computer modelling and simulation to enable a person to interact with an artificial three-dimensional visual or other sensory environment. VR applications immerse the user in a computer-generated environment that simulates reality through the use of interactive devices ...”.

This definition dictates that VR based environments must be three-dimensional either to the eyes or to other senses of the human body. Other similar definitions go further by requiring specific levels of this perception such as the use of goggles or other means of depth cueing.

Another aspect of the Britannica definition involved immersion. This concept appears frequently in other definitions such as one by Steuer [63] which requires

that a VR based environment must make the user feel, to a certain extent, that they are present in the environment. This is referred to as *presence* [38], the concept loosely defined as a user's subjective sense of 'being there' in the VE.

Most definitions, like the Britannica definition, mention both technology and immersion factors, although some may focus on one and totally disregard the other. With so many definitions, some systems may be considered VR by one definition and not by another. For this research, we clearly need to select a definition. As a result, we choose to proceed with the simplest definition and regard VR as a "medium in which ideas can be communicated via the use of a computer-simulated virtual environment". In the next section we look in to how VR, as a medium, can provide advantage over actual reality and use this to further define what VR is.

2.1.2 Virtual Reality as a Medium

Virtual Reality is a communication medium in the sense that it facilitates transfer of information or content from one party (the author) to another (the consumer or the end-user). VR is a relatively new medium if one were to compare it to other well known or well-mastered media such as the written or spoken word. VR as a medium has an advantage over traditional reality-bound media in that it can make use of imaginary or non-reality-bound knowledge to convey the message, and yet still present it in a familiar format to the user. This allows authors to use various combinations of the two types of knowledge, both reality and non-reality bound, in order to create the best VE for the job.

The differences in the combinations are reflected in VR applications. Here we look at how the different types of VR applications take advantage of each type of knowledge.

1. VR Applications Taking Advantage of Knowledge of Reality

Most VE authors try to approximate reality by making their VE look, act and/or feel like reality. While other media present information that needs to be interpreted mentally, VR allows presentation of a reality directly to one or more of the users' senses in a form that requires little processing. As such it calls for no extra training in the media because a typical user already possesses the necessary knowledge – the knowledge based upon the perception of a common reality. In this sense, VR is similar to the film medium. Sherman *et al.* [58] described how for both media, content consumers rely on their own real world experiences, meaning that there was no need to acquire special skills for effective use. However, this is only the case if the virtual environment mimics reality in some ways. To use the film example, the viewer is able to understand and appreciate a clip better if they are able to relate to it. However, unlike the film medium, VR allows for even closer mimicking of actual reality by allowing interactions between the user and the medium. We think that it is this high level of closeness to reality which the various definitions are

imposing on a VR system i.e., they insist that a VE be as close to reality as possible.

The emphasis on technology translates to how close **physically** (the look) a virtual environment is to the real environment. The 3D goggle introduces an extra depth, bringing it closer to reality. Data gloves enable users to provide input in a way they are used to in reality.

Similarly, the emphasis on the level of immersion translates to how close **mentally** (the feel) a virtual environment is to the real environment. This factor encompasses other non-physical factors contributing to the level of likeness to reality. The notion of *being there* is a way to insist that at some level the user is able to relate to the environment – that it is real to them. In short, in order to take advantage of the existing knowledge of reality, virtual reality needs to be as close to reality as possible, both in terms of look and feel.

Applications taking advantage of this attribute of VR are various simulators that strive to be as close to reality as possible. They are often used for a variety of purposes where performing the task in real life would be too prohibitive in some ways, including:

- **Cost:** Examples are flight simulators where the costs of training on a real aircraft may be prohibitive. The United States National Aeronautics and Space Administration (NASA) makes use of VR technology [83] for training of astronauts for space-based tasks. Other examples include remote surgery [19] and communications systems such as remote meetings.
- **Danger:** There are a number of applications that have been designed to reduce the risk of harm to the user. For example, it is safer (and also less costly) to train new pilots in a flight simulator. Remotely controlling robots via a VR system [65] can be used to accurately and safely operate in hostile environments such as in war zones.
- **Physical Possibility:** VR applications are being used to create environments which may not be visible to humans. An example is medical simulation of molecules, such as the HI-virus for educational purposes [77]. Some VR applications also have the advantage of repeatability and precise control of the virtual environment. An example is the use of VR to teach physics by Brelsford [11] which allows student to change the earth's gravitation pull and observe the changes in behaviour of a virtual pendulum. Yet another example is the use of VR in psychology to attempt to cure patients of various phobias by repeatedly exposing them to increasing levels of their fear in order to desensitize their response [64].

2. VR Applications Taking Advantage of Non-reality-bound Media

The VR medium, not being reality-bound, has a distinct advantage. It allows for the creation of an environment not possible in reality. After all, what

would be the point of using virtual reality if only reality-bound media were allowed?

Applications falling into this category include:-

- **Visualisation:** Like two-dimensional charts and graphs, VR allows for the presentation of data [17] in a more efficient and usable way. VR applications have been used to aid visualisation of many data types including scientific [49], databases [43] and archaeological [1].
- **Design Tools:** VR has been used in rapid designing and engineering of complex objects [49]. They provide effective means to do this by employing non-realistic, non-immersive techniques.
- **Games:** While there are computer games which strive to be realistic, many provide entertainment value by putting users in imaginary worlds.

The two types of knowledge (reality-bound and non-reality-bound) form opposing forces. Taking the advantage of non-reality based knowledge often means that the environment loses its close matching to actual reality. In some ways, mastering the VR media involves striking a balance between making use of reality bound and non-reality bound knowledge. Figure 1 illustrates this idea. However, using non-Reality bound media does not have to overly affect the level of realism in the VE if done correctly. For example, often an author may choose to integrate well known actions that are not possible in reality such as flying or walking through walls.

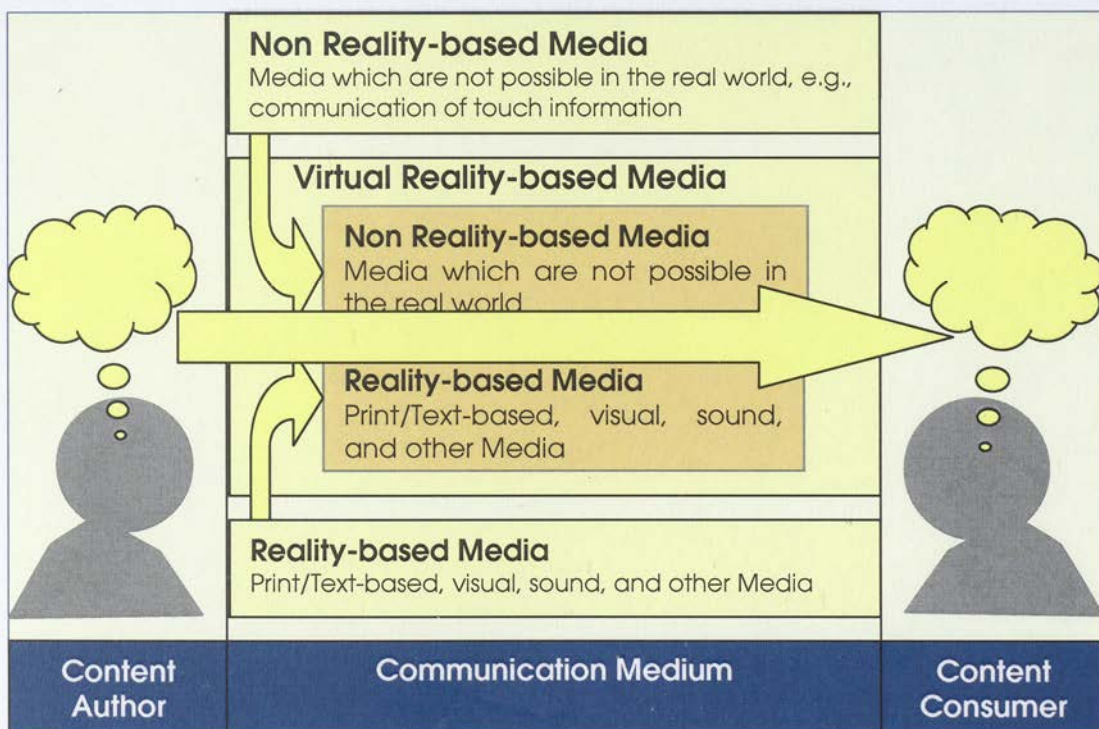


Figure 1: Virtual reality as a communication medium - a product of reality-based and non-reality based media.

In summary, the main advantage of VR is that it is based on a common reality, while at the same time allows the author to use other means not possible in reality, to convey the content across to the content consumer. However, it is a balancing act to make sure that by including actions which are impossible in reality, they do not lose the linkage to reality of their VE.

Thinking of VR as a medium, we define virtual reality as follows:

Virtual reality is a communication medium in which technology – usually a computer – is used to simulate the perception of reality in the mind through the sensory inputs, taking advantage of

- *The user's experience with actual reality or realities to achieve better transferral of information.*
- *The medium's ability to provide non-reality bound media for use in transferral of information.*

With this definition in mind, we refer to the content which is presented using the VR medium as a virtual environment.

2.1.3 Effectiveness of VEs

In this section we look at the effectiveness of VEs and use this to infer effectiveness of VR as a medium. Firstly, there are many different ways in which the effectiveness of a VE can be measured. Vastly different types of measures have been introduced. Traditionally, one may measure the level of closeness to actual reality. For example, one can examine the technology and measure how closely the produced environments resemble reality. Another way to measure the level of closeness, is to examine users' feeling in the environment i.e., the idea of immersion or *being there*.

Yet another way to measure a VE's effectiveness is to evaluate its effectiveness in transferring the intended information to the consumer. This can be done simply by measuring the accuracy of information before sending and after receiving of the information.

As stated in the previous section, VR as a medium allows for more accurate transfer of content from the author to consumer by taking advantage of the fact that both have a more or less common view of reality. However, this assumes that the author is able to package the intended information easily and effectively for the medium. The medium may facilitate the conveying of message to the consumer, by relying on the user's perception of reality, but there is nothing to help the author to virtualise actual reality. This aspect of VR (the creation of a VE) can negatively impact on the effectiveness of any created VE. If this is done properly, the rest of the communication medium will transfer the information correctly by design. Therefore, the effectiveness of a VE could potentially rely on how accurately and effectively the author is able to package his information for the medium.

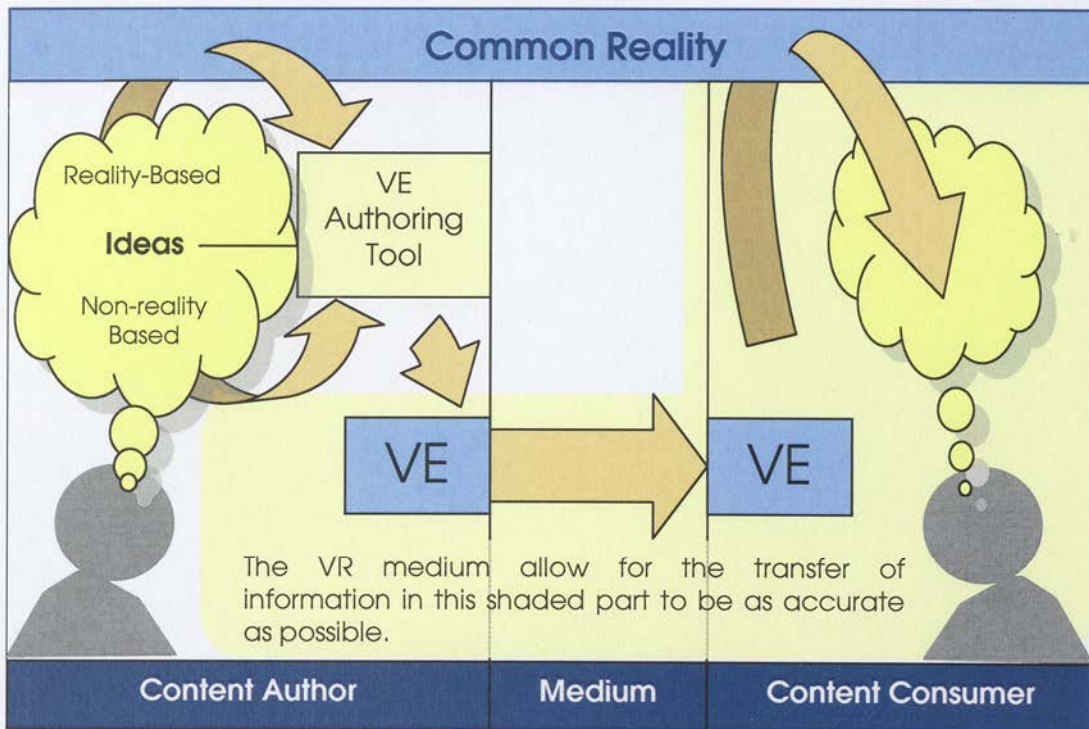


Figure 2: Authoring for the VR medium: While packaged information in the VE may be accurately transferred from the content author to the consumer, there is a chance of a mistranslation from the author's idea into VE.

As Figure 2 illustrates, an author's idea is a mixture of reality and non-reality based ideas which need to be incorporated in to a VE using an authoring tool. The resultant VE is then interpreted by the user by basing it upon their knowledge of actual reality. There is potential for miscommunication of the author's idea due to other factors including:

- **Bad knowledge of reality:** It is true that people live in a common reality and up to certain level share similar perceptions of reality. However, when authoring for a specific application, there is a chance that the author's, or the end-user's, knowledge on the subject is insufficient. For example, if an author is trying to create a VE for a flight simulator but has no knowledge of the required engineering and flying skills, the created VE may not matched the reality experienced by pilots. In order to avoid this scenario, the author must have sufficient knowledge of the subject upon which the VE is based.
- **Bad usage of non-reality medium:** When an author uses an idea which is not based on actual reality, there is a chance that the consumer may not interpret it as intended by the author. This is a question of using the media correctly and it is the authors' responsibility to identify which techniques work for the VR media. In time, VR authors will have developed guidelines on which techniques are effective.
- **Bad translation in the authoring process:** This happens during the authoring process in which the author is turning ideas into a VE. This is

common with most other media and is the main potential cause for miscommunication.

The effectiveness of a VE thus depends on both the author and the authoring tools they use. There is a need to further develop techniques which authors can use to take advantage of the VR medium, but like other media, this happens over time as more people make use of the media. There is also the important factor of knowledge of the relevant subject on which the VE is based. Both of these factors mentioned thus far can be acquired by the author, but the effectiveness of the authoring tool is beyond the author's control. They are bound by what the tools can do for them. This is the reason we proposed this study. We would like to have a look at authors who have sufficiently mastered aspects under their control and how they have coped with available authoring tools. The authors in questions are those who have sufficient knowledge of the subject at hand; perhaps having communicated it successfully in other media. They must also have adequately explored the VR media. We refer to this type of authors as *content-expert VE authors*.

2.2 VE Authoring and VE Authoring Tools

By treating VR as a medium, its effectiveness is dependent on the efficiency of the VR to convey messages. Since an authoring tool acts as an access point to the medium, the authoring tools and processes become very important in creating an effective VE. In this section, we discuss different aspects of these tools and processes. We begin by defining the two types of parties involved in the VR medium:

Authors: The creator of the virtual environment and thus the user of the virtual environment authoring tools. As stated in the previous section, our intention is to focus particularly on content-expert authors and in our text, we mainly refer to them as *authors*.

Users: the actual end-user of the virtual environment. Typically, the VE user exists in the virtual environment as a three-dimensional object called an *avatar*. In our text we will often refer to these VE users as *end-users* or *users*.

2.2.1 Virtual Environment Authoring Methodologies

Methods of creating a virtual environment are evolving and researchers have suggested ways in which this could be done. This include the methodology proposed by Kaur [35], involving five phases, with the first four being planning stages.

1. **Requirements Modelling:** The purpose of the VE needs to be clearly established.
2. **Conceptual Modelling:** The process of collecting real world data such as images, sound, sketches, storyboards, etc. These data could then be used as a basis for creating the virtual world.
3. **Perception Modelling:** The author identifies the different feelings which the VE should induce in a user.
4. **Structural Modelling:** This stage involves the modelling of VE structures, such as the design of scene graphs and behavioural models. These models may or may not map directly to the virtual environment depending on the constraints of the VR system.
5. **Building:** The final phase where the VE is created.

A large proportion of Kaur's methodology involves planning with the actual process of building the environment taking up the last step. Fencott [23] and Smith *et al.* [62] also advocated devoting a large amount of time to the planning phases, followed by a once-off authoring phase. Figure 3 illustrates the link between the different stages of Kaur's model.

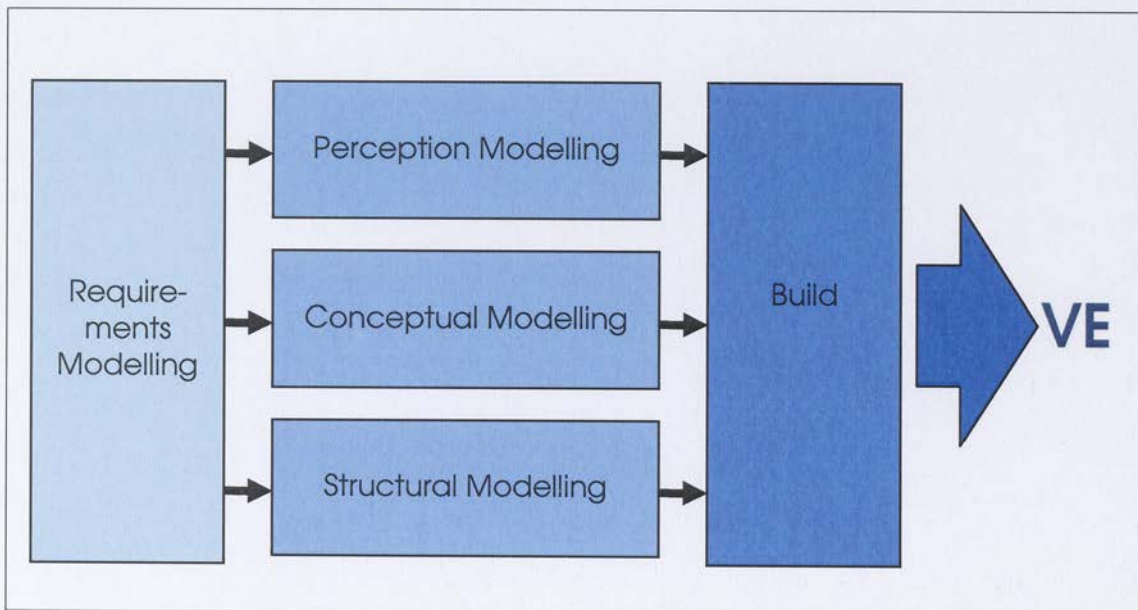


Figure 3: Kaur's Stages in creating a VE

2.2.2 Virtual Environment Authoring Tools

In this section we examine tools which are used in a virtual environment authoring process. These are tools which have been designed to aid in certain aspects of the building phase outlined in the previous section. They can be classified according to the tasks which the author needs to perform.

1. **Modelling Tools:** These tools are used to create new or modify existing three-dimensional objects. This process can be equated to creating the virtual building block for later building of the environment. Many tools provide basic objects such as spheres and cubes which can then be combined, or transformed into more complex objects. Examples of such tools are 3D Studio Max [69] and Maya [70].
2. **Static Environment Building Tools:** These tools allow authors to place and arrange three-dimensional objects to their specification. The output of these tools is often the *shell* of the virtual environment with no possible interactions. Usually, the static environment tools allow for some basic object-level manipulation such as scaling or changing of colours and/or textures. Example of such tools are GenEdit¹ [76] and FloorPlan 3D [75]. It is also possible to use the basic modelling tool, such as Maya to create the basic structure of the VE. Furthermore, some tools also provide basic interactions for certain objects. For instance, FloorPlan 3D has built-in *open* and *close* interactions for doors and windows.
3. **Interaction Specification Tools:** These tools complete the creation of a VE by allowing the author to specify interactions between objects in the environment, including that between objects and the VE user. Often these tools are based on a programming language and have to be linked to the VE in some way. For example, *Visual C++* [86] is typically used in conjunction with Genesis 3D [76] to create a fully interactive world. Scripting languages, such as *Python* [81], are also employed in some tools including Alice [71], Virtual Environments Program [85] and WorldViz's Vizard VR Toolkit [87].
4. **Other Facilitating Tools:** These are other tools which are used by the author to facilitate the authoring process. These include tools like image manipulators that are used to create textures and sound-editing software packages.

Some currently available tools falls into two or more categories as they seek to reduce the number of tools involved in the authoring process. Many tools are also packaged together so as to create better compatibility between the different functionalities. There have also been efforts towards a single authoring tool which allows authors to perform all the necessary tasks in any order they choose [89]. These four types of tools fit into Kaur's stages of VE authoring as shown in Figure 4.

¹ A environment editing tool for Genesis 3D.

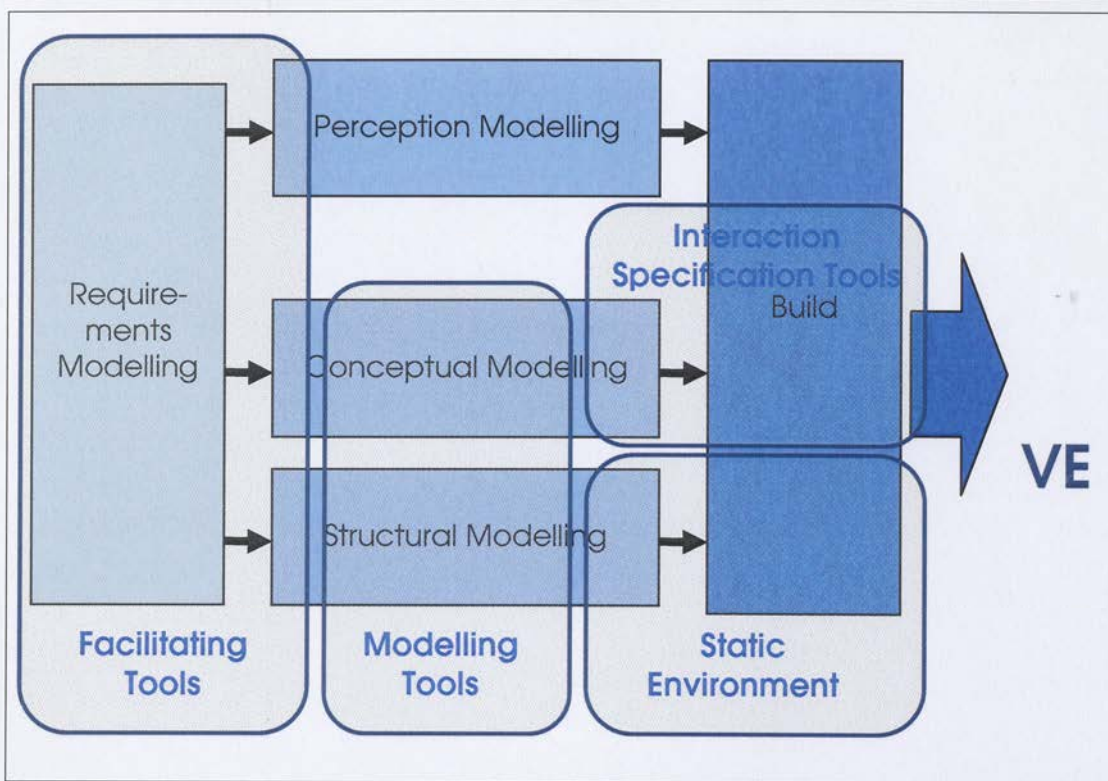


Figure 4: Existing types of tools and how they fit into Kaur's stages of VE authoring

2.2.3 Authoring and Display Platforms

All the authoring and supporting tools mentioned thus far have been based on two-dimensional desktop platforms. However there is a much wider range of display platforms for VR. Many in use today are classified according to the level of technology on which they are based. This factor also influences the level of immersion of the VE created for each system. The categories are as follows:-

Two-dimensional display systems: These systems, also known as desktop-VR, are based on a two-dimensional projection of three-dimensional environments displayed on a flat monitor. Usually the users provide input by means of keyboard and mouse, mostly without any motion tracking.

Systems with stereoscopic depth cueing: Stereoscopic depth cueing methods are employed in these systems to create extra depth, making environments closer to reality. This is over and above other depth cueing techniques (such as occlusion) that are already typically employed on two-dimensional display systems. In some cases, users are required to wear a pair of stereoscopic glasses that extracts images for each eye, thus creating a three-dimensional appearance. In other cases, a pair of small but high resolution screens are employed; with each presented to each of the user's eyes. This type of system is found in head mounted display (HMD) devices. Some systems may also make use of a motion tracker, mainly to track the position and orientation of the user's head.

CAVE [15][16][17]: This is the one of the most advanced VR display technology involving projections of images on multiple surrounding walls, floors and ceiling. It was developed at the Electronic Visualization Lab [74]. Together with the use of stereoscopic glasses and motion tracking, users can be totally immersed within a fully three-dimensional and interactive virtual world.

The difference in platforms may play a significant role in affecting the author's ability to convey their messages using the VR medium. Most traditional media authors works on the same platform in which the resultant media will be viewed. An example of this is a painter – the canvas being worked on will be the one the audience will see. While the advent of computers has made it a more than capable media to author on, such as in the publishing industry, the popularity of the WYSIWYG interface for publisher software seems to indicate that the closer the *canvas* is to the end-result, the better it is for the authors.

2.3 Summary

In this chapter, we have presented various definitions of VR. Some of these are dependent on the VE display platform, while others include a description of how the VE should make the user feels while using it. The different ways of testing a VE's effectiveness are often dependent on these factors. However, we took an alternative view of treating VR as a medium and evaluating its effectiveness on how successful an author is in conveying his or her message across to an end-user. Looking at VR this way means that the authoring tools and processes become crucial as it is the authors' means of communication.

We also stated our intention to specifically focus on content-expert authors. These are authors who have mastered communicating their ideas and message using other media and are now looking at exploiting the VR medium.

3

Relevant Research Methodologies

In this chapter, we present a brief overview of appreciative inquiry (AI), the methodology that was employed in our study. AI was based on action research and has been used successfully in many projects, mainly in social and corporate contexts. We looked at different aspects of AI, including application aspects that might need tweaking, as well as some criticisms of AI that have been expressed.



3.1 Action Research

3.1.1 Introduction

Action research [5] is usually an iterative process which allows action, in terms of improvement and change, to be affected while simultaneously furthering the understanding and knowledge of the subject in question. Originally proposed as a triangle of action, research and training, action research has been applied in many fields. The action research of today is very flexible and is typified by close involvement with subject experts in a cycle of:

1. Detailed assessment/review of the current process and desired change
2. Proposal and implementation of change to be made

Often the above cycle is repeated until a satisfactory level of change and/or research has been reached.

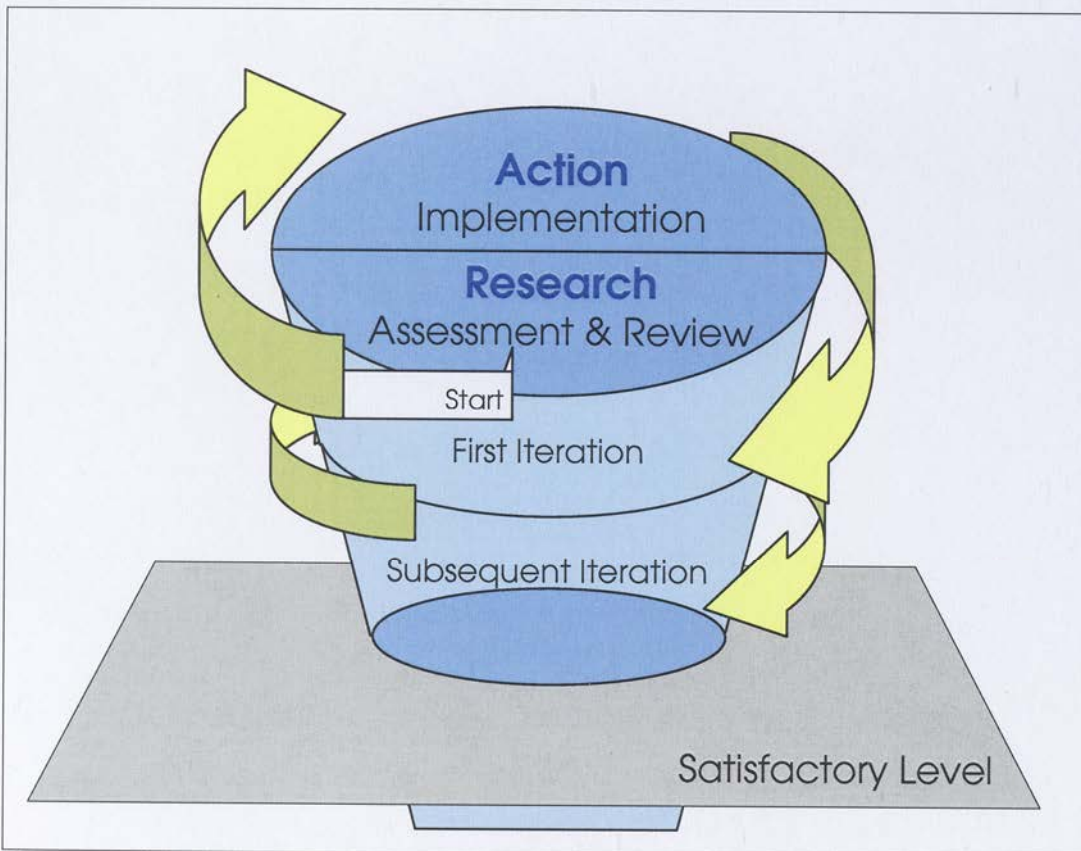


Figure 5: Task Cycles in Action Research

3.1.2 Action Research in Information Systems Research

Although action research has its roots in social and medical sciences, it has been applied in various ways in information systems. Typical methods used in published information systems research involve some form of participatory action research [6] which starts off by defining a client-system infrastructure. This involves agreement between the researcher and the domain experts and subsequent specification of the environment or domain within which the research will take place. The process is then followed by five identifiable iterative phases [3][4]:

1. **Diagnosis:** Identification of primary problems causing or leading to the demand for change.
2. **Action Planning:** Discussion and proposal of desired direction for positive change and solutions which may effect this change.
3. **Action Taking:** Collaborative implementation of proposed changes.
4. **Evaluating:** Collaborative evaluation of change against the desired direction.
5. **Specified Learning:** Essentially the knowledge processing step resulting in either improvement of the process or elimination of a failed idea.

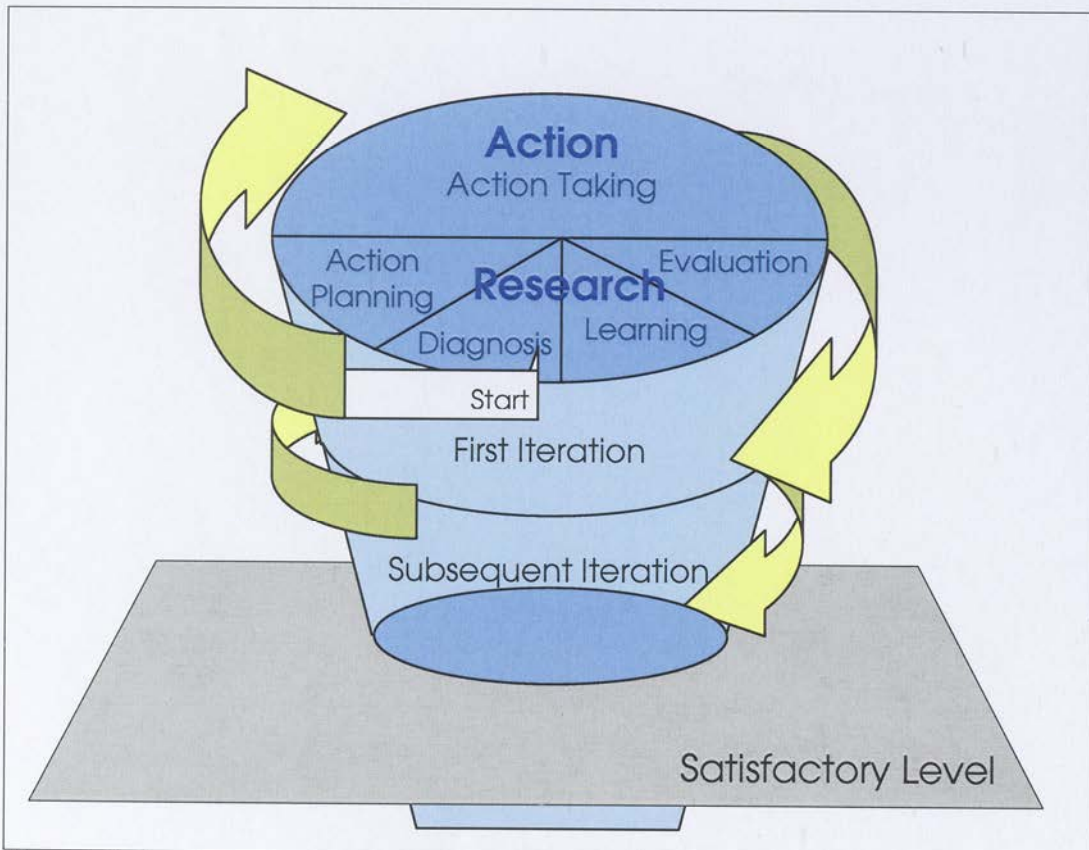


Figure 6: Task cycles in action research in the field information systems

Action research, by its very nature, has its goals redefined every iteration. Thus it might never finish (unless an overall goal has been specified beforehand). Many researchers have found it to be valuable even when executed in a fixed number of iterations. A single iteration often produces valuable insight and solutions. This improves with subsequent repetition.

One of the main differences in this application of action research is the starting position. In the original method, a complete iteration does not result in the evaluation of the work done in that iteration. Instead, this is left to the next iteration. In the version employed by most information system researchers, the research phase is split midway and the iteration ends off with the evaluation. This allows for iterations to be independent of one another.

3.2 Appreciative Inquiry (AI)

Appreciative inquiry is a positive approach to reorganising a process. It is often used to improve an existing process, even when it is not obviously problematic. It was initially developed as a derivative of the conventional forms of action research [14]. The AI process was developed by Dr Cooperrider and his colleagues at the Weatherhead Graduate School of Management at Case Western Reserve University [72]. There has also been research into AI and how it works and how best to employ it [21].

Reflecting its action research roots and its positive nature, a typical AI takes place in four phases, often known as the four Ds cycle [54][67].

3.2.1 The four Ds Cycle of Appreciative Inquiry

1. **Discovering** *the best of what is*: Finding out and appreciating the positive aspects of the process or organisation under study and the people involved. There will also be many negative points and opinions. Although AI advocates remaining positive, these opinions should not be ignored.
2. **Dreaming** *of what might be*: Thinking about the desired or ideal direction in which to take the process or organisation. The information from the previous phase, including the negative points raised, is used to derive the *dream*. These should be presented in a positive sense. For example, when a bad way of doing something is discovered, participants are questioned on how they think it would be best to do that particular task. The point of this phase is to find out the direction to take.
3. **Designing** *what should be*: Subsequent collaborative design of ways to achieve the dream. The point to keep in mind is that people often change their mind as they think about the problem more. It might take some time and many visits to arrive at the actual solutions. In most AI project, only a team of people (e.g.

village leader, management and works representatives) is involved in the design process.

The design phase's main purpose is to realise the dream envisioned during the previous phase and ground it in reality. There may be many factors which were not considered during the discussion in the Dream phase.

4. **Delivering the dream:** In the final phase where various stake holders act out the proposed new design. Recently this phase has been referred to as the *destiny* phase as an emphasis that all parties should be involved in order to continue the momentum of contributing positively.

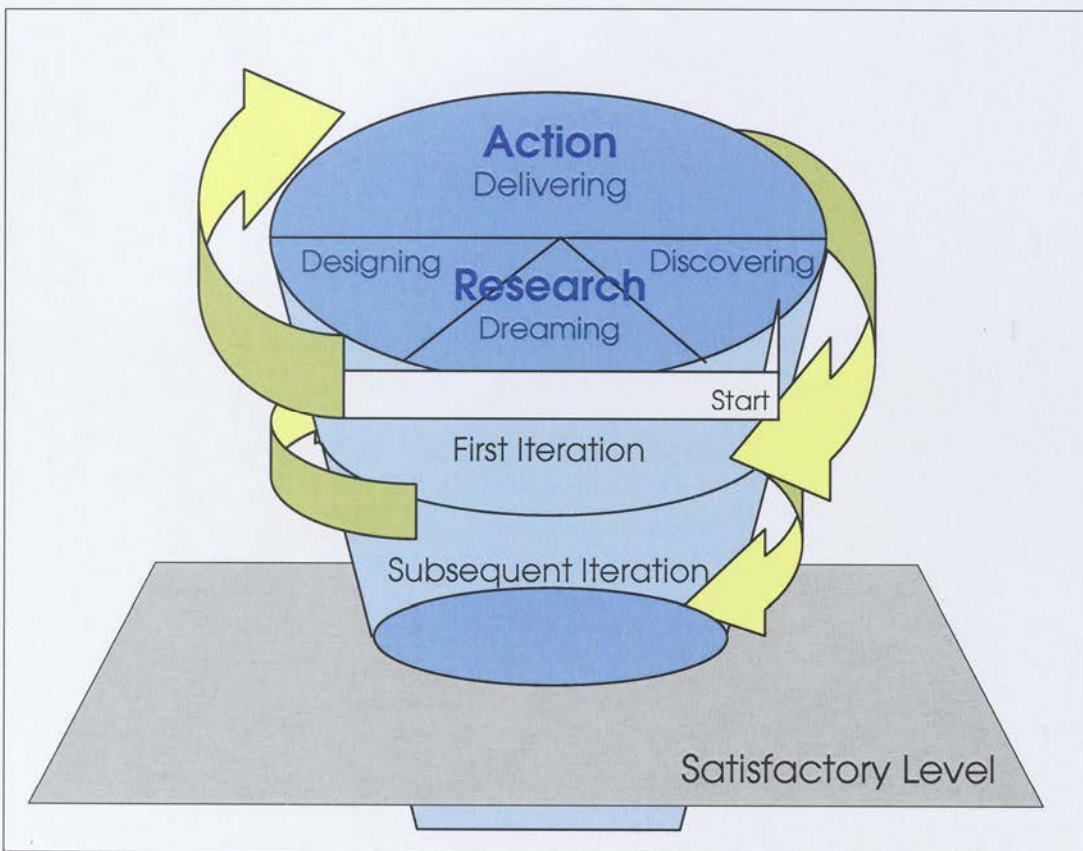


Figure 7: Tasks cycle in appreciative inquiry

Appreciative inquiry follows the original action research format starting off with a research phase followed by the action phase. The evaluation of the current iteration is left over to the discovering phase of the next iteration.

3.2.2 Positive Energy

Appreciative inquiry's main strength is its *appreciative* aspect of relying on a positive approach to the discussion and involvement of all parties in order to achieve betterment for all. This appreciative approach is reflected in all four phases which

emphasise staying positive and turning negatives into positives. There is no blame attributed to any parties, but rather a positive attitude towards overcoming the problem by suggestion of ideas based on positive experiences. These aspects of AI make it effective in a social context where there might be the potential for conflicts. By focussing every stake holders' positive energy, progress can be made [21]. The other main aspect of the inquiry is the close involvement of the researchers and the people that are part of the process in question. At the heart of the inquiry are the interview sessions which researchers are urged to approach positively. Some interview guidelines have been suggested [29][30][21][55][56] including:-

1. State everything in the affirmative and avoid the negative. Avoid questions along the line of "Why didn't you ... ?" .."isn't it bad that .. ?"
2. Start off with a broad definition of the topic to make sure everyone is on the same page.
3. Ask probing questions such as "Tell me about the best experience you had with [topic]" Present questions as an invitation for the participants to open up and start telling their story. This can be done by using expansive, open-ended and positively-worded questions. Ambiguous questions can be useful, but be honest. Follow up with further questions to get to the root of the positive energy.
4. Keep a caring and appreciative attitude. Listen to answers attentively with interest. Believe participant's answer rather than doubt. Make it a conversation not an interview. Some researchers go as far as not calling it an interview. Try to have a conversation; do not try to put on an expert role.
5. Make a note of any negative comments and rather come back to the issue at the end. However, if the participant is passionate about the issue (and keeps coming back to it), let them get it off their chest before continuing.

3.2.3 Underlying assumptions

Although appreciative inquiry demands a positive frame of mind, it assumes that the topic under investigation is simplistic and can only be viewed from a small number of viewpoints. The research is often required to make the naïve assumption that every participant is in a co-constructive relationship with each other. Some researchers [54] in the area point to this naïveté of the methodology as a strength, as it allows researchers to correct this point of view as time goes on. It is not effective where issues are not clear cut, or where there are multiple issues with conflicting demands. In order to make use of AI, researchers need to make sure that all the stake holders are in a co-constructive relationship or that communities are broken down to areas where such relationships are prevalent. Often diversity in the participating groups contributes significantly to the success of the exercise. If everyone involved has the same viewpoints there may not be a positive viewpoint for overcoming an issue. This is because AI works by spreading good ideas for consideration and later incorporating some into the existing system.

3.2.4 When to use AI

AI has been recommended for use in problem-solving or to drive change in many areas, but the following can be used to gauge the suitability of AI to a particular process [21][29]:

1. **Changes are required, but individual levels are already working:** AI has proven to be effective where overall change is needed and the participants know things could be better. They know where they fit into the process and that they perform their job well in that area. However, the problem may lie with other areas they are not familiar with, such as how the sub-processes work together. In these cases, traditional problem solving methods may have been employed without success, mainly because the problems do not lie on the individual level [21].
2. **Conflict situation:** AI is also effective where there is potential for conflict or if the participants or subgroups are confrontational towards one another. This is because AI works by not focusing on faults and blame.
3. **No one speaks out or everything is perfect:** In cases where no one wants to speak out or where at an individual or subgroup level, participants are happy. There are cases where everyone says nothing is wrong but the overall picture tells a different story.
4. **Common Goal:** For AI to work, everyone must have same goal, in the sense that what they are doing is geared towards achieving the same thing. This is echoed in AI's underlying assumption of co-constructive relationships.
5. **Participant Diversity:** Diversity in the participating groups is very important. The larger the pool of people the more likely and innovative the solution that could be attained.

3.2.5 What do with Negative Data

Talking a positive approach in an interview does not necessarily lead to only positive data. Participants often treat an interview as a complaint sessions. Even with utmost care, many negative comments will be made. Different ways have been suggested to deal with this negative data [56], while still keeping within the framework of AI. This includes deriving positives from the negative. An issue identified as negative reflects the absence of an issue in the positive ideal image. This can be used in the dreaming and design process to improve the current process.

Excessive negative data can also indicate a problem in the relationship between the stake holders or even in the underlying assumption about the co-constructive

relationship. In extreme cases, it might also indicate an inherent problem in the process itself – often a radical change might be required.

3.2.6 Determining the End

AI is an ongoing process by design. This is evident in its lack of a separate evaluation step to end off the cycle. For most of its applications in social areas (for examples see [21][28][29][30]), the process were never meant to end. The idea is that in every iteration, factors affecting the process change (e.g. new challenges like natural phenomena such as drought). The process must be ongoing to deal with these challenges in order to achieve some vague objective (e.g. making optimum use of what is available). However, increasingly there is often a clear objective to be achieved or a date when the process has to end, especially, where there are budgetary considerations, for instance, in corporate or governmental organisations.

3.2.7 Evaluation

Van der Haar and Hosking in [67] expressed concern about how AI can be objectively evaluated given its focus on appreciation and lack of a prescribed evaluation step at the end of each iteration. An evaluation of sorts is done at the initial discovering phase where it would either appear as a positive aspect or not appear at all if it were to raise a negative issue. At face value it seems that AI may potentially absorb bad or negative ideas which may not be discussed at all after its inclusion.

Evaluation of the design² has been incorporated in some AI-based projects in the form of positively-focused interviews, similar to the initial discovery interviews. This is done both during solution implementation (though in this case it could be seen as part of the design phase) and at the end, just before the delivery. The results of the interview are then considered in the next iteration, in the discovery phase [28], allowing the process to end at this point.

3.3 Summary

In this chapter, we introduced the appreciative inquiry (AI) methodology that has been employed successfully in many areas of social and corporate worlds. It is a continual process that focuses on positives to achieve a better process. However, AI is not without its problems. At face value, focusing on the positives seems to encourage people to forget about the negatives. AI also does not prescribe set ways to evaluate, but some projects have employed successful positively focused evaluation discussions. Nevertheless, we feel that AI is an exciting methodology for user-interface studies, the reasons for which will be discussed in the next chapter.

² Evaluating that it is in line with what came out of the dream phase

4

Methodology

In this chapter, we discuss our application of appreciative inquiry in our study. We start off by explaining our reasons for choosing AI and how we went about applying it. Although AI is an exciting methodology that has proven successful in many contexts, we felt that some of the issues discussed in the previous chapters warranted some minor modification of the methodology.

We also present the different processes and participants involved in our study and how they fit into the modified AI model.



4.1 Adaptation of Appreciative Inquiry

4.1.1 Why use AI?

In many ways, our study is very similar to other problems that have been solved successfully using AI. Comparing our situation to those outlined in subsection 3.2.4 **When to use AI**, we find the following points concerning VE authoring that would make the usage of AI suitable:

1. Content-expert authors considered their work satisfactory, but most have a feeling that things could be better. They might not be readily able to identify issues or problems with the way they currently do things. Perhaps the problem lies with the user interface of their tool. They are experts in neither VE authoring nor user interface design. This fits well into the '**Other change required, but individual level already working**' point of when to employ AI.
2. Traditional user interface research is prone to the 'demand characteristic' [51] whereby participants in the studies try to please the researchers by tending to give what they consider to be what the researchers want to hear. Additionally participants might not want to appear *stupid* by indicating any problems. This is the case of '**everything is perfect**' response that AI has worked on successfully in the past. These are cases where participating parties insist that everything is well as it is. There is an example of a similar situation that was solved by AI – the merging of two secondary schools in the Netherlands when both insist that they were working optimally already. However, the overall problem of the merger could not be dealt with by consultants until AI was employed [21].
3. Content-expert authors have the similar goal of creating an effective VE for any given purpose (**Common Goal**). Although the tools they use may be different, their goals are essentially the same. The variety of interfaces can contribute to the **Participant Diversity** requirement of using AI. They will provide a pool for existing good ideas that can be employed to solve any problems that surfaced.

These points contributed to our confidence that appreciative inquiry is suitable for our study. However, there are some aspects of AI that we need to modify to suit our study.

4.1.2 Detail of Adaptation

We adapt appreciative inquiry to our studies by looking at how the original action research methodology was adapted for the information systems field. This leads to our version of appreciative inquiry reflecting more of the action research ideas, where appreciative inquiry has its roots. The following adaptations were made:

1. We will use positive and negative aspects of the interview to contribute to both the dream and the evaluation phases. We expect that at least some comments would be negative and that these should not be lost in the process. Since these data will be used in deriving potential solution, we expect there to be no adverse negative impact on the deployment of the methodology.
2. We will also make sure that the proposed changes are in fact in line with solving the issues. This will be done via the evaluation interviews. In typical AI deployment, these changes would be re-evaluated at the beginning of the next cycle. However, these would not be looked at in terms of issues identified in the previous cycle. It was felt that linking the issues to the solutions would provide better evaluation. As such we expect this modification to not result in any negative impact by this modification.
3. In a usual application of appreciative inquiry, the entire population of participants is available to take part in the exercise. However, this will not be possible in our case, and as such we choose to apply appreciative inquiry to a small sample of the VE authors and to then verify that the results of the studies applied to the wider population of VE authors via a verification survey. There have been other cases where a small number of individuals from a population (e.g., village leader) took part in the study. AI has been found to work well in these cases, even though there were no attempts to make sure that the participants form a representative sample. In our case, we will make sure that the authors are from a diverse background. This, in addition to the verification survey, will hopefully indicate to us that the selected sample is in fact representative.
4. We will also employ multiple cycles of feedback and redesign in the design cycle. This already occurs informally in a typical deployment of AI (see the description of the Design phase in section 3.2.1 The four Ds Cycle of Appreciative Inquiry). As such, we expect there to be little or no potential impact of this modification.
5. Due to time constraints, only one cycle of AI was performed. However, this should be sufficient to bring out the most pressing issues. It should also be sufficient for us to gauge the suitability of applying AI in a study such as ours.

The following steps were taken. Refer to Figure 8 for an illustration on how they fit together.

1. **Identification and Recruitment of authors:** Identification and recruitment of authors from various applications of VEs.
2. **Discovery Interviews:** Discussion with selected authors to identify positive aspects of their experience as well as issues concerning user-interfaces in current VE authoring tools.
3. **Consolidation Interviews:** Interview sessions for consensus between different parties on which issues are relevant or more important.

4. **Design Cycles:** A series of discussions and prototyping to arrive at solutions for each problem. The discussions were with the same group of authors from task 2.
5. **Evaluation Interviews:** A different group of authors from similar backgrounds took part in independent reviews of each solution. These authors were involved in neither task 2 nor task 3.
6. **Verification Survey:** A survey was conducted to confirm that the identified interface issues also affected VE authors as a whole, and not just those from the selected application areas. The survey was to confirm that the solutions were applicable and did not create further problems in other VE applications.

In the rest of this chapter, each task is discussed in more detail.

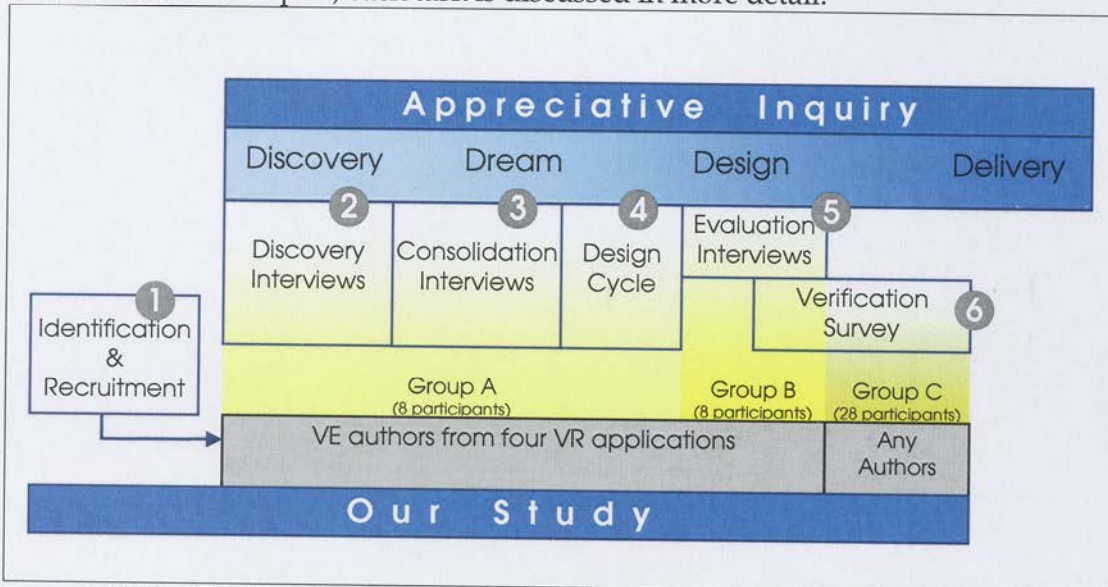


Figure 8: Methodology employed in our research – an adapted version of appreciative inquiry. The discovery and consolidation interviews formed AI discovery and dream phases. Part of the consolidation interviews also contributed as the starting point for the design process. The evaluation process and verification process acts as an endpoint to the design process and also as a kick off point to the delivery process by introducing our solution to the general population group. In total three different groups of VE authors took part; refer to **4.2 Identification of authors** for more detail.

4.2 Identification of authors

A number of VE authors participated in our project. There were three main groups, referred to as Groups A, B and C. In all three groups, the following profile was used to select suitable authors:

- **VE authoring Experience:** They should have at least six months experience in developing VEs. This requirement was to make sure that they had spent some

time trying out different methods and, as a result, developed for themselves ways to create different aspects of a VE.

- **Tools:** The authoring tools used should be readily available and not obsolete. This was to confirm that the tools used were not only in use by a few peoples. We accepted a tool specifically designed for specialised content-expert authors; provided that it was not still under development and that it was available for us to have a look at if the need arose. The tools used should also be able to produce sufficiently complex virtual environments. There is a tendency for VE authoring tools targeting novices to restrict the complexity of the VE which can be authored.
- **Tools Experience:** Four application areas were selected. These were architecture, education, entertainment and research. The main reason for choosing them was to make sure that they bring in diverse experiences in authoring tool usages. From informal discussion with potential participants, it was discovered that architectural creation of VEs mainly involved the environmental building task. On the other hand, educators' use of VE authoring tools mainly involved interaction specification. Creation in entertainment and research were balanced in these two areas.
- **Learning Process:** They should have learnt to use the tools themselves, but those attended a short introductory course were not excluded, provided they had learnt to apply the tool to their application themselves. As our approach depended heavily on these authors, it was felt that authors participating should not have been too influenced by particular authoring method which might have been taught to them during intense and specialised training courses.
- **Own Industry Experience:** They should have extensive experience in their job (usually associated in some ways with VE authoring) and have been using VEs as a tool to the benefit of their products or company. Furthermore, they should also have created at least one suitable environment that the author or others in the industry have considered to be effective in the tasks or goals initially set out. This set of requirements was to make sure that the author creates useful VEs and that the VE creation was not just an exercise with no commercial purpose.

The make-up of each of the three groups was as follows:

- **Group A:** Twelve authors from four areas (architecture, education, entertainment and research) were approached and identified as suitable. Of that, a pair from the four application areas (making up eight participants in the group) agreed to take part for a 12-18 month period, in which they would be available for consultation at least one hour per 6-8 week period. In total, there were eight participants in Group A.

Group B: Participants in this group were recruited at a later stage when the development cycle had been completed. They were used mainly to

independently verify the effectiveness of the solutions in addressing the interface issues. There were eight people in Group B from the same four application areas as Group A.

Group C: Participants were recruited at the same time as Group B. Participants from Group C were asked to anonymously fill in the online *Verification Survey*. This survey is outlined further in **4.7 Verification Survey**. In all, forty-four people completed the electronic survey but only twenty-eight fit the required profile outlined above. These were the only results which were evaluated. Participants from Group B also filled in the survey; their results were kept separate for further analysis.

Figure 8 also illustrates each group's involvement in the study.

4.3 Discovery Interviews

The purpose of the discovery interviews was primarily to catalogue the experiences of content-expert VE authors as they learnt to use the tools. The interview followed the appreciative inquiry guidelines outlined in the previous chapter. Both positive and negative comments were recorded, with negative ones followed through at the end of the interviews as per the guidelines. Each of the eight participants from Group A took part in this phase.

Each interview session took, on average, 75 minutes to complete and focused on the following areas.

- **Confirmation of the suitability of participating authors:** This was to verify that each participant met all the requirements described earlier.
- **Explanation of the study:** The purpose of our study was explained to each participant. They were also informed of the approximate timeline of when they would be needed. This was so that they have some understanding of what was required of them and when.
- **Discussion of general interface issues:** In each interview, discussions on the current set of tools used by each author were held. This part of the session was held with the appreciative inquiry guidelines in mind and focused primarily on the user interface issues. This was the main purpose of these interviews and the majority of time was spent doing this. The interview sessions were loosely structured into three parts.
 1. **Walk-through of own authoring process:** Each participant took us through their authoring process, mentioning anything that came to mind, but were told that we were particularly interested in how they learnt the tool, how effective they thought the tool (or certain feature of the tool) was, any features that they would like added or changed, and how. They were also

free to mention anything else they thought might be of interest while taking us through the authoring process.

2. **End-product VE discussion:** We had a look at some of the end-products of the authoring process. From this we were able to discuss the nature of the resultant VE, as well as any issue concerning effects on the VE by the tool and whether this was as a result of the interface, the author, or the functionality of the tool itself.
3. **General discussion:** The remainder of the interview sessions were spent discussing any other comments or issues that authors wanted to bring up.

During this interview, it was made clear that there was no time limit and that should they come up with any relevant issues concerning their experiences in VE authoring, they were told to contact us about it and further interviews could be arranged. Seven participants contacted us with further issues but most were relatively minor and were included into the study without further interviews. However, two further interviews were arranged.

As a result of the discovery interviews, a list of issues was compiled for each participant. Positive aspects were then compiled for later use in the design process.

The lists were then compared to one another for the negative aspects. Some problems were essentially the same but were worded differently; some were subsets of another. A final list of issues was then compiled with each ranked according to how many participants mentioned it. This was to enable us to prioritise issues since we were limited to a time constraint. In the end, we ended up with fifteen issues that were mentioned by at least three participants. The issues were grouped together into a bigger set where appropriate. We ended up with a list of fifteen interface issues.

4.4 Consolidation Interviews

The discovery interview sessions yielded a listing of a wide variety of issues. Most of the consolidation interview sessions were spent on discussions concerning how much better their tool could be on the various issues. Most participants were kind enough to spare approximately 2-3 hours for this phase.

The interviews consisted of the following tasks.

1. **Reminder of issue:** The session started off with a reminder of the issue under discussion and the reason why the study was undertaken. This was done to refresh their memory.
2. **Presentation of interface issues:** We presented the list of interface issues and clearly indicated to each participant where and how their previous comments fitted in. When necessary, they were given further explanations of the issue they were having difficulties with. They were then asked if they agreed with

each issue and were invited to give any suggestions or criticisms on the compilation.

3. **Rating of issues:** Participants were then asked to rate each issue according to how much they thought it affected the three factors, using the scale of 0 to 4 (0 – none, 1 – low, 2 – medium, 3 – high and 4 – very high). The three factors were: authoring process, learning process and overall quality of the resultant VE.

As a result of these interviews, a list of fifteen interface issues, ranked according to their average severity, was compiled. Out of twelve, nine issues have an average rating of 2 or higher. These nine issues were selected as candidates to be addressed in the development cycle phase.

Figure 9 presents the summary of the discovery and consolidation processes as outlined earlier in this section.

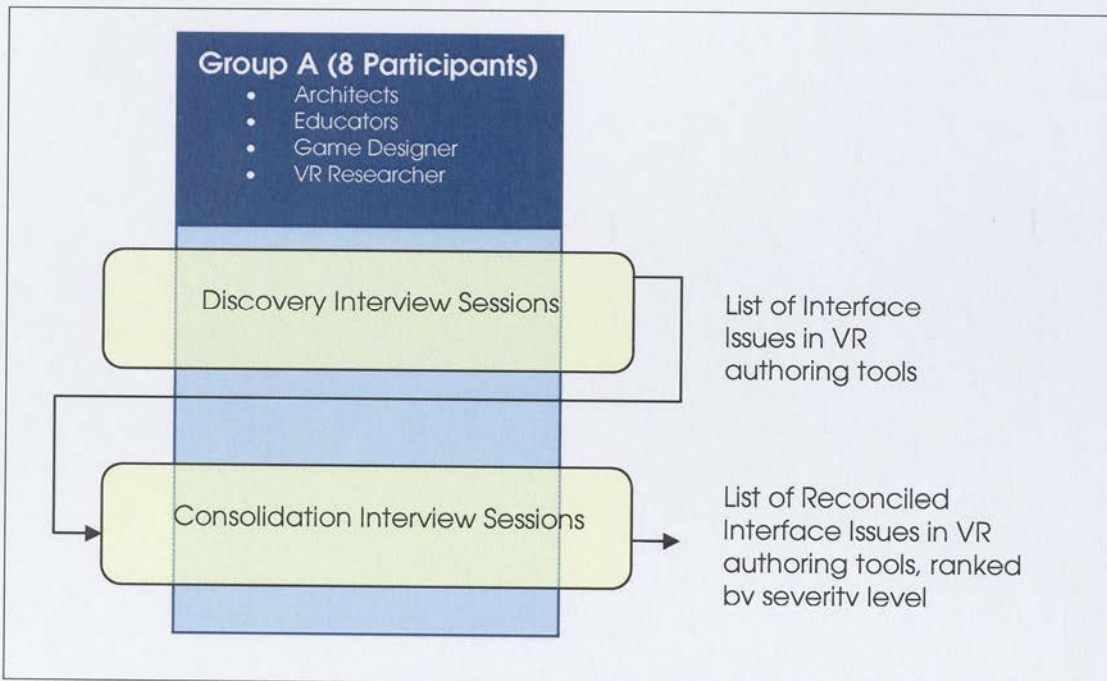


Figure 9: Summary of discovery and consolidation phases

4.5 Solution Design Cycles

The purpose of this phase was to arrive at solutions which would address the list of issues from the previous phase. During this phase, we worked closely with participants who had initially identified these issues (Group A). These solutions were then evaluated on their effectiveness in the next phase.

The process of arriving at the final solution to the interface problems involved research and consultation with all participants and a prototype implementation. We were very

careful in making sure that solutions did not introduce further problems. This was done by taking into account other interface issues on the list and that the solution would not impinge on other solutions or aggravate other issues.

From the background of problem description and rating information from the consolidation interviews, the following tasks were taken in development of each solution. The three steps were repeated as necessary, such as in cases where two potential solutions were proposed.

1. **Research:** This included investigating the user interfaces of existing tools and academic publications on related topics. This formed the starting point.
2. **Discussion and brain-storming:** In these sessions ideas were presented to each participant. They were:
 - Ideas that were derived from positive comments or proposed by other participants. Some reconciliation was necessary since solutions may work for one group of authors but may create problems for others. For example, the level of accuracy of object placement necessary for architectural purposes could result in an unnecessarily complex interface for other authors.
 - Ideas derived from current research. However, some of this research was found to be unsuitable for various reasons³ and, as such, some were not considered or rejected out of hand by some participants.
 - Ideas which we have derived. Having looked at the problems, we proposed ideas that we thought might be an effective solution.
3. **Prototyping:** The most suitable solutions to the issues (as agreed by all involved) were prototyped. They were presented at least once to each participant for comments and criticisms. These were incorporated where possible into the prototypes. The final prototypes were then shown in the last sessions where any further comments were documented. At this late stage, most participants were happy with the prototypes. For some ideas, only interface mock-ups were produced, especially in cases where behind-the-scene functionality was not required.

³ For example, all interaction specification interfaces involving advanced programming were rejected.

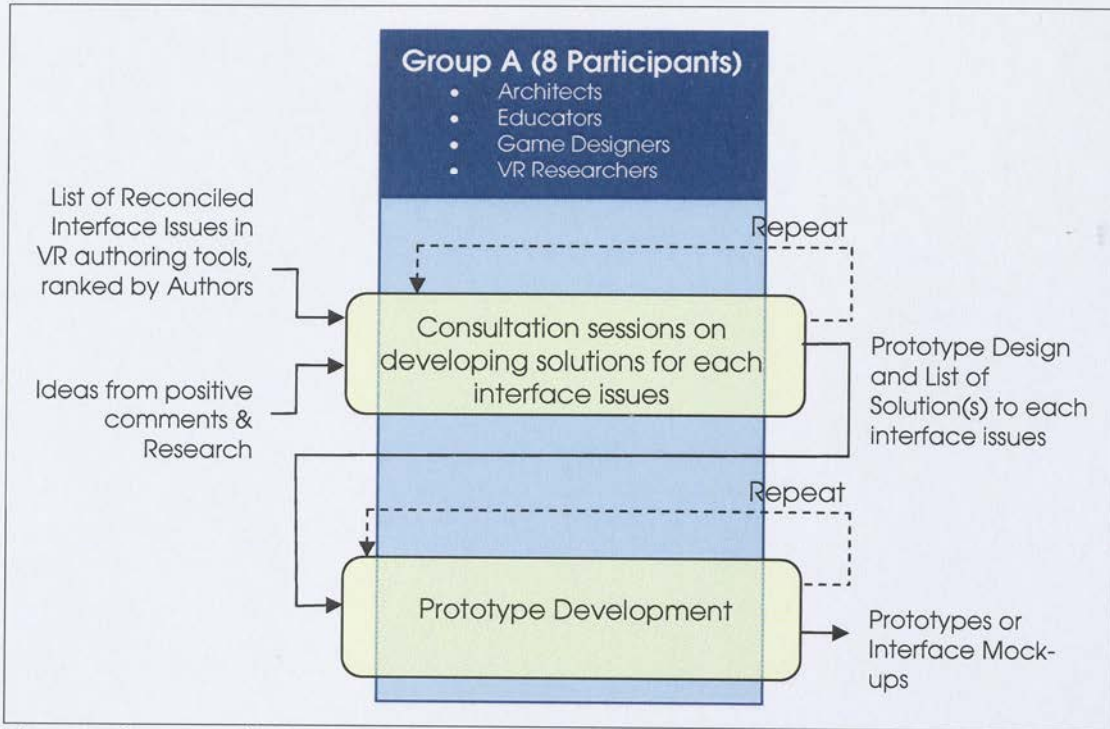


Figure 10: Summary of Solution Development Cycles

4.6 Evaluation Interviews

The purpose of the interview is to independently assess the level at which each solution addresses its corresponding issues. This was done by means of **Hands-on evaluation interviews** involving eight participants. It was not possible to have hands-on evaluation with more people due to unavailability of participants willing to take part. The participants were from the same fields as those in the problem discovery process, but they were not involved in that process.

Although the authors who took part in the problem discovery interviews and solution development phases were satisfied that the solution alleviated each corresponding problem, they were not part of the evaluation process. It was felt that having been so close to the development process they would be inherently biased towards each solution. It was for this reason that a different group of authors were used in the evaluation process to independently assess the effectiveness of each solution. The purpose of the hands-on evaluation interviews was two-fold. Firstly, it was to confirm that each participant experienced similar problems as those from Group A. Secondly, the interviews were used to assess the level to which each participant felt the prototyped solution had solved, or at least alleviated, each issue.

Each interview was conducted as follows:-

First phase (30-60 minutes)

1. A brief discussion on user interface design was held to ensure that the participant had some basic understanding of what constituted interface

issues. It was discovered during prior discussion sessions that some authors did not understand this concept.

2. Concepts behind each of the solutions were explained and demonstrated and explained to each author. They were given the opportunity to ask questions and to play around with the demonstration prototype or interface mock-ups. They were then asked to comment on the interface and how each might affect their authoring experience. If the author touched on issues that were linked to the solution under discussion, this was noted down, and the issue was discussed further.

Second phase (30-60 minutes)

The discussion went from one interface issue to another in random order; with each interface issue the following steps were taken:

1. Authors were presented with a brief explanation of each interface issue. They were asked what they thought of the issue and the effect it had on their authoring process, learning process and overall quality of the resultant VEs.
2. We presented the relevant aspects of our solutions. We then discussed the solution and the level that he/she felt the set of interface solutions contributed towards the issue resolution. We also discussed ways to improve our solutions and any potential problem they saw with them.

4.7 Verification Survey

The purpose of this survey was to confirm that the issues identified during the discovery and consolidation interviews, and their subsequent prototyped solutions, apply to the wider VE authoring community and not just the four application groups taking part in the process. However, participants from Group B also filled in the survey and their answers were kept separate. The survey was conducted anonymously online. Participants were not timed, and had the option to stop the survey after a page was completed and resume at any point afterwards. The web server retained the information on how far a participant had completed the survey. Additionally, once a page was complete, participants could not return to that page. The full survey is available in Appendix C and consists of the following sections

Questions 1-4 inclusive: Questions asked in these pages were to determine participants' experience in VE authoring, their authoring process, the VEs that were created and the tools used. The answers were used to decide whether or not to include the participants in the analysis. If a participant did not qualify, the survey continued, but their results were not included in the final analysis.

Questions 5-15 inclusive: Each interface issue was divided into four separate pages. Some issues were broken down into small issues for the sake of clarity.

1. The issues were presented and participants were asked if it was present in their authoring tool. If it was, the participant was taken to 2, if not then they were taken to the next set of questions concerning the next issue.
2. On this page, participants were asked questions concerning the nature of each issue. Answers for these questions were used to confirm our analysis of the issue.
3. This page presented questions on the level at which the issue was problematic during both the initial learning of the tool and the authoring process. If the participant found the issue to be problematic to any of the two processes, then they were taken to 4, otherwise they were taken to the next set of questions concerning the next issue.
4. The final page briefly outlined the proposed solution and asks for participants' opinions on how much effect this would have on their authoring ability.

Figure 11 presents the summary of the evaluation and verification processes.

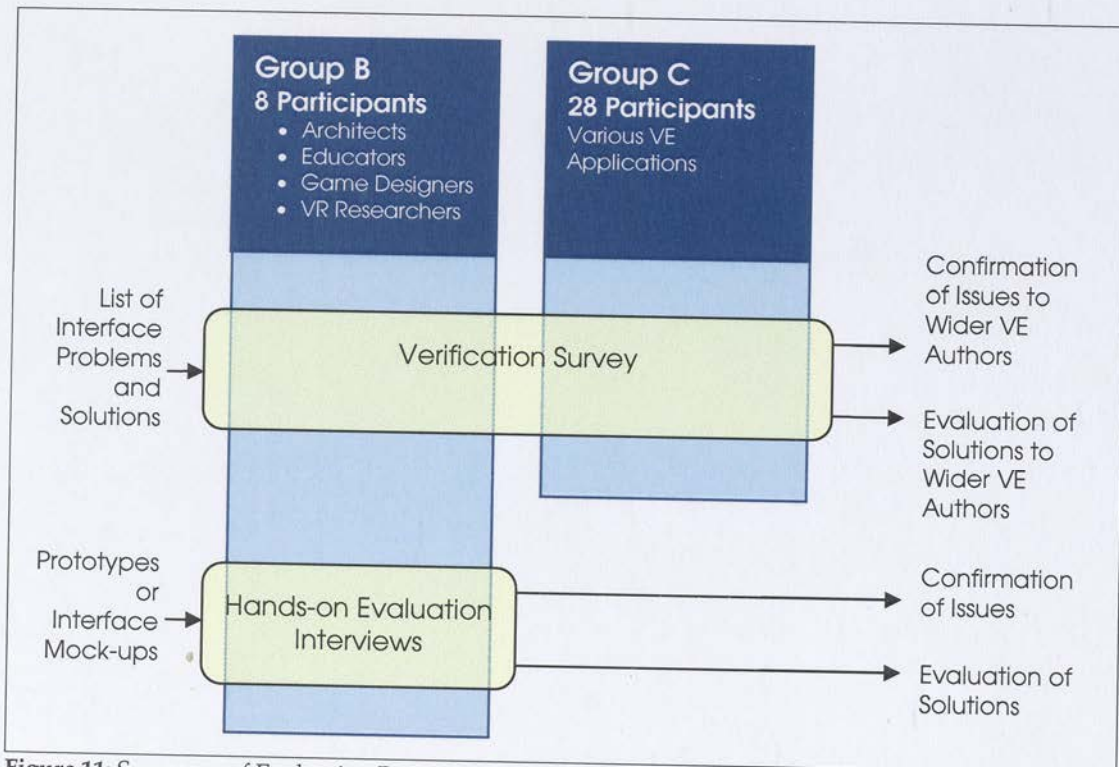


Figure 11: Summary of Evaluation Process

4.8 Overall Methodology

In summary, we have described the methodology which was employed in reaching effective interfaces for VE authoring tools, aiming specifically at content-expert authors. Initially, we worked with a group of eight VE authors from four areas of VE applications. The discovery phase was done with these authors by means of a series of interview sessions. From this a list of interface issues was compiled. Solutions to these issues were arrived at with collaboration from the eight authors; this led to prototype or interface mock-ups integrating these solutions. The prototypes and interface mock-ups were then evaluated by a different group of authors from the same application areas, in order to independently verify their effectiveness. To confirm the existence of these interface issues (and the effectiveness of each solutions) in the wider VE author population, an online anonymous survey was conducted.

Figure 12 summarises the overall methodology employed in our study.

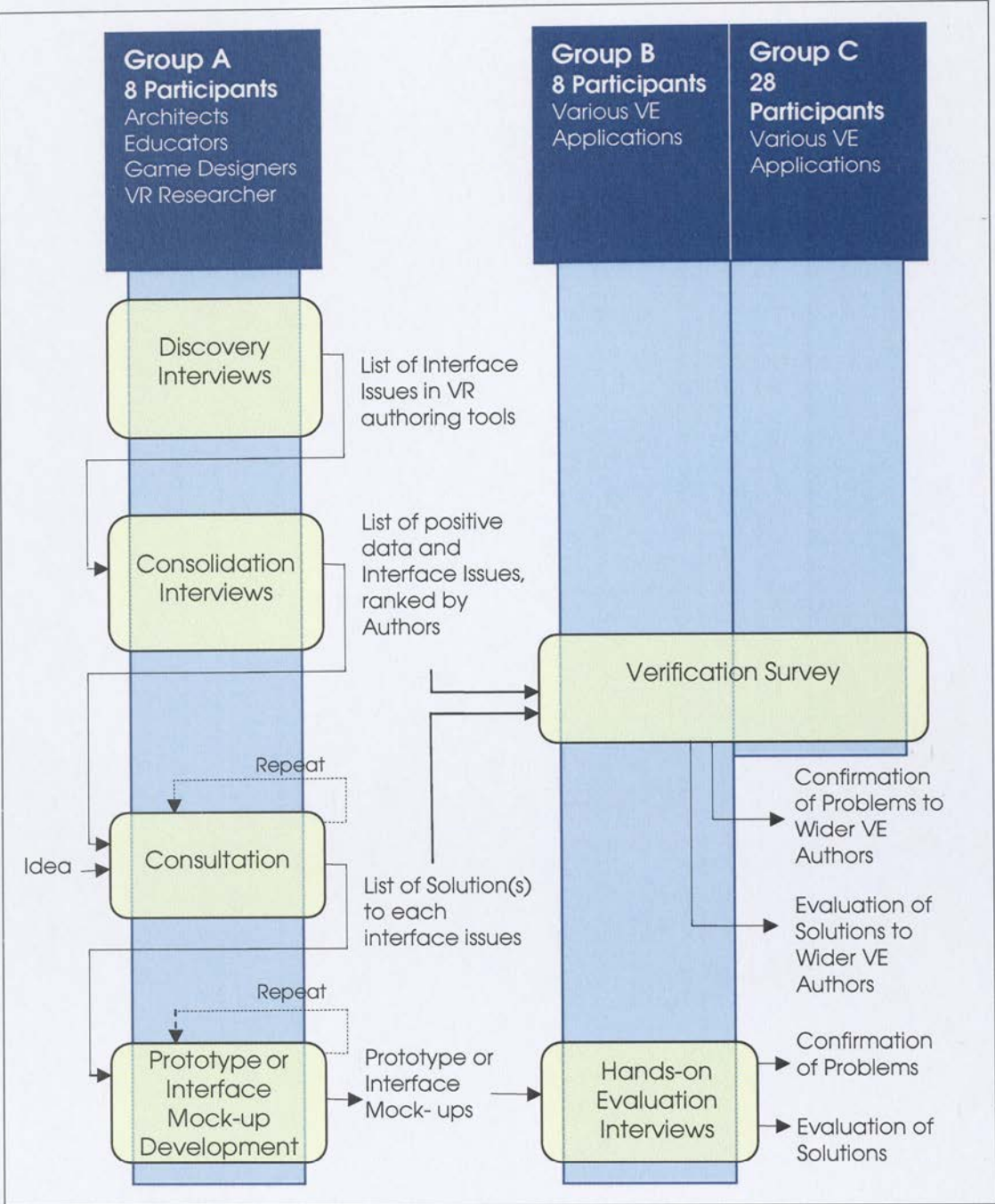


Figure 12: Overall Methodology

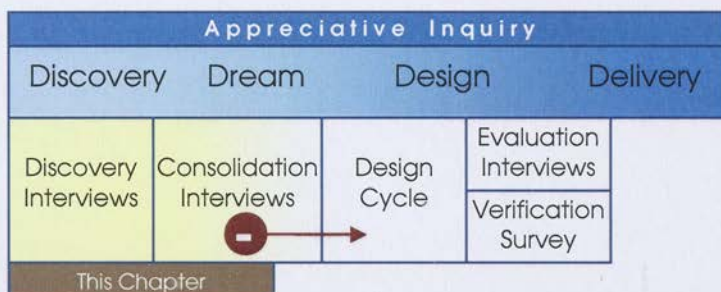
In the next few chapters we will be discussing the different processes and their respective results as follows:

Appreciative Inquiry			
Discovery	Dream	Design	Delivery
Discovery Interviews	Consolidation Interviews	Design Cycle	Evaluation Interviews
			Verification Survey
Chapter 5		Chapter 6	Chapter 7



5

Interface Issues



In this chapter, we present the main results of the discovery and consolidation interview sessions. The two interview sessions were part of the discovery and dream phases in the appreciative inquiry methodology. In our modified version of AI, however, negative data were not discarded but instead used to form the basis of the eight interface-related issues presented in this chapter. (Note that positive data from this interview were used during the design process, the results of which will be presented in the next chapter.)

The eight issues were identified in the interviews of a group of relatively experienced virtual environment authors. We also present their comments as well as a brief analysis of possible reasons as to why the interface issues were integrated into current tools.

These comments and analysis, together with effects of each issue on the authoring process, learning time and overall quality of the virtual environments, will form a basis for solution development to be presented in the next chapter.

This chapter is broken down into eight sections, each concerning one interface issue that was identified by a group of relatively experienced virtual environment authors. Each section is broken down into subsections as follows: -

Description of issue: A brief description of the issue. For later discussion each issue will be given a number and will be referred to as P_x - where x is the issue number.

Comments on the issue: summary of authors' comments on the general consequences of the issue. The sections also include comments on the specific consequences of the issue in three key areas, authoring process, learning process and overall quality of virtual environment produced. General comments will be labeled with $P_{x_{cy}}$ - where y is the y^{th} comment made with respect to issue x , while specific comments will be labeled with $P_{x_{c/AP}}$, $P_{x_{c/LP}}$, $P_{x_{c/VE}}$ for comments concerning authoring process, learning process and overall quality of VE respectively.

Analysis of issue: Analysis of the origin of the issue - possible reasons why it was integrated into some tools. Each analysis will be labeled with $P_{x_{Ay}}$ - where y is the y^{th} analysis made with respect to issue x .

From this chapter onwards, we will be using the above notation to refer to issues, comments and analysis.

5.1 Multiple Tools (P1)

Authors were forced to master at least two tools, usually one for environment building and one for interaction/behaviour specification. Similar problems also exist with a single tool that has two sets of interfaces or modes for each of these tasks. For all these tools, the two tasks are done independently and one task cannot be done while the other task is being completed. Figure 13 illustrates the process involved when multiple tools or interfaces are used.

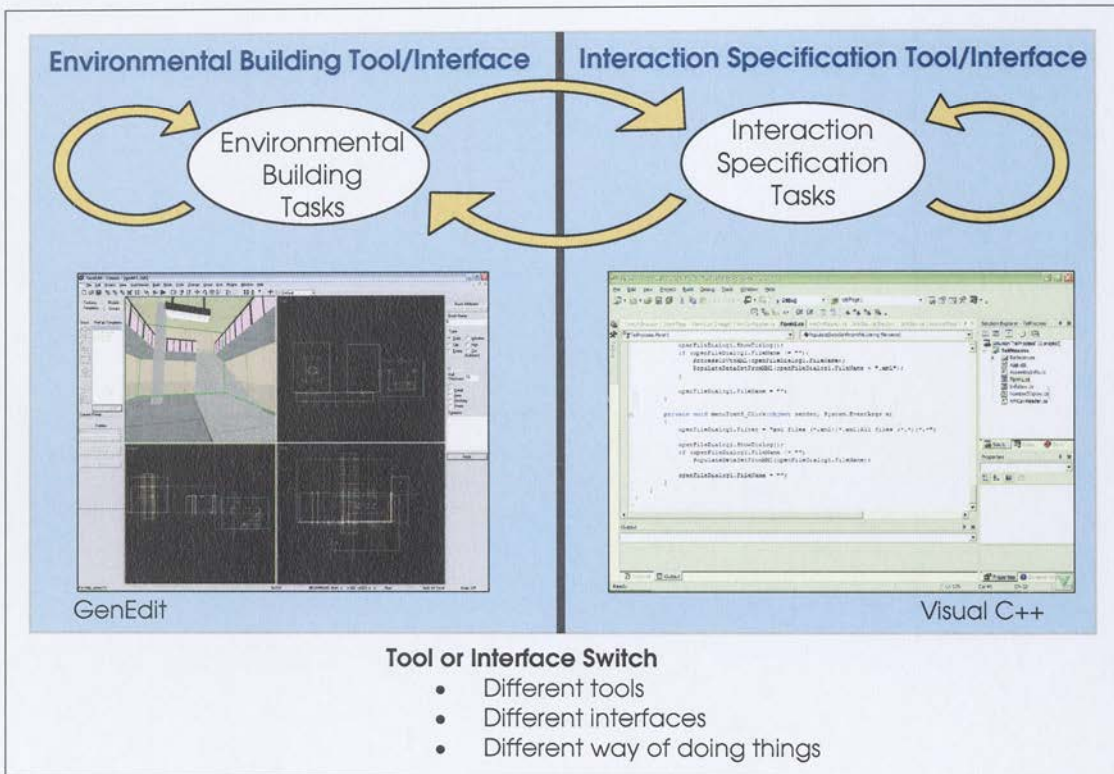


Figure 13: VE Authoring with multiple tools or interfaces

There are several examples of tools with this issue; the ones encountered during our study can be classified into two categories as follows:-

1. **Two or more tools, each for different tasks:** For example, one author, on building a VRML VE, uses 3D Studio Max to perform the environmental building task; while using a crude text editor to independently code the interaction. Similarly, another author uses Microsoft Visual C++ IDE to code and later import the interactions to the VE.
2. **One single tool with different modes for each task:** Most tools falling into this category merely group together different tools that are not tightly integrated. For example, an in-house tool, (BUILD) used by an architect provides access to a library of 3D objects which can be added to a VE. It also provides an interface to an editor which is invoked when the author wants

to specify crude interactions, such as opening doors on a visitor's approach. After coding such interaction specification, the author must accept the change and then manually link the interaction to the VE by means of a button click.

All of those interviewed during the initial phase used multiple tools during the authoring process. Most used a wide variety of tools which were not part of a single package. There are some, however, that have attempted to provide a *single box* solution. In the survey, thirty out of thirty-six participants (83.3%) indicated that they too used more than one tool or tools that provided widely different interfaces or mode for different tasks.

5.1.1 Comments

P1_{C1} Confusion about the different tools

During the interview, authors have reported becoming confused about which tool is being used and for what purpose. For example, authors often become frustrated at having to learn which actions or tasks can be performed in a particular mode or with a particular tool. Even the more experienced authors indicated that at times they were unsure about the tool best suited for a task, often because the tasks could be performed in many ways. A popular example is that moving and manipulating the physical appearance of objects is not possible while in a tool, or mode, for defining interactions, even though objects can be manipulated physically on screen. Question 5.2.1.4 in the Verification Survey queried the participants on this issue. The result, illustrated in Figure 14, showed that about half the authors experienced this issue during VE authoring process.

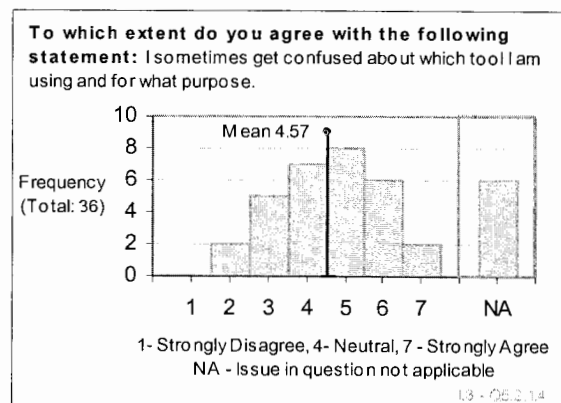


Figure 14: Question 5.2.1.4 from the survey which enquired authors about confusion in using multiple tools. (NA indicates users not affected by the issue.)

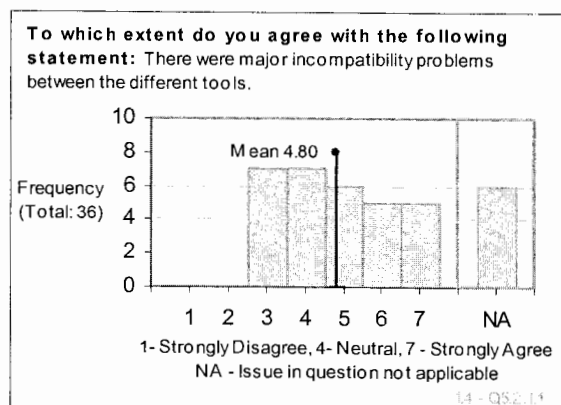


Figure 15: Question 5.2.1.1 from the survey which enquired authors on incompatibility aspect of using multiple tools. (NA indicates users not affected by the issue.)

P1_{C2} Incompatibility issues

Multiple tools used by some authors created major compatibility issues. These ranged from data format

incompatibility to fundamental issues such as colours and sounds that are different across the tools. Some participants also indicated that this was an issue for them as well, as indicated by Figure 15.

The cause of these incompatibility issues could be that these tools were not tightly integrated, in effect making the user perform a complete task with one tool, save their work to a particular format, then reopening it with another tool to complete the task. Figure 16 shows the result of the survey concerning integration. Although most participants indicated that the multiple tools they used lacked integration, there was a sizable portion which indicated that their tools were sufficiently integrated. As expected, most of those participants experiencing incompatibility issues also indicating that the tools they used were not tightly integrated. The reverse was also true.

P1c3 Multiple steps Mentality

Perhaps the most problematic aspect was the suggestion that two separate steps were to be performed while authoring a VE. During interviews, authors often separated authoring steps along the line of the tools' capabilities. This is especially true if the author had been exposed to only one set of tools. Most authors (83% of those surveyed – see Figure 17) thought of the authoring process as involving two separate steps – building of the static environment and interaction specification. Furthermore, Figure 18 shows the trend that most authors using multiple tools tend to think that VE authoring is a two-step affair.

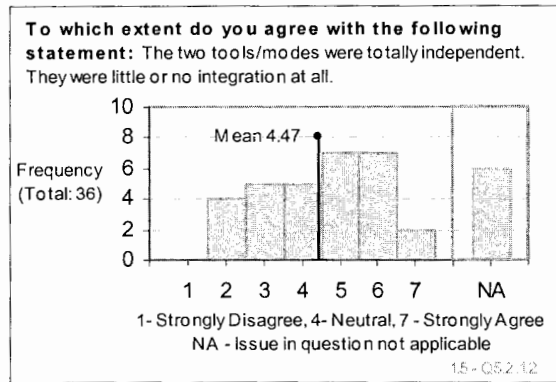


Figure 16: Question 5.2.1.2 from the survey which enquired authors on the non-integration aspect of using multiple tools. (NA indicates users not affected by the issue.)

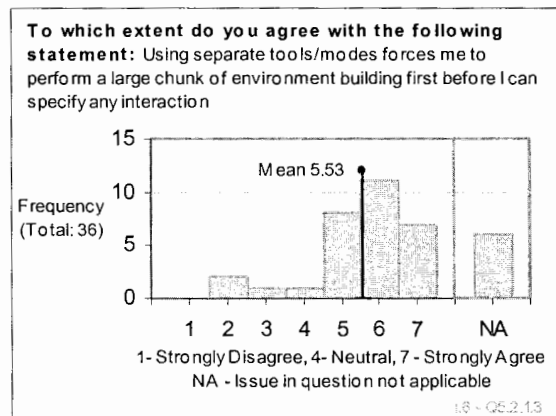


Figure 17: Question 5.2.1.3 from the survey which enquired authors on multiple steps mentality of using multiple tools. (NA indicates users not affected by the issue.)

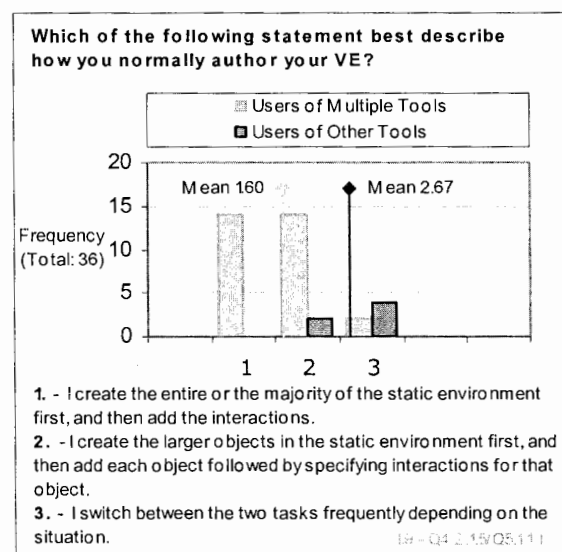


Figure 18: VE building habits, grouped into those using multiple tools and those that do not. (Results from PS Survey Question 4.2.1.5, cross-referenced with Question 5.1.1.1)

Many interviewed authors became used to performing tasks along the line of the functionalities of the tools or mode, so much so that they often found it difficult to adapt to other tools or packages. Although some saw this as authoring style, it makes it difficult to change to a newer tool that might force a different way of doing things - new ways that might actually be more flexible or effective. This effect was very profound in two of the participants who had at their disposal newer version of the tool they were using. Even though the newer tool was more flexible and no longer forced the build-save-code mentality, their attempts at using the newer package adhered closely to this mentality.

P1_{C4} Unnecessary saves or backups

In many cases, authors were forced to save to disk before changing to another tool or mode. For some, this added another level of complexity, forcing authors to keep track of the filenames used and the changes made in each tools. This often led to errors.

For most authors, this created the sense that they had to save often and rely on it to act as a form of undo functionality. While saving often, in itself, may not be purely negative, authors often lost track of changes and some ended up with many outdated versions of the environment. In survey

question 5.2.1.5 (Figure 19), participants were asked about this issue, and while most authors did indeed feel that they were forced to back up their VE every time they switched tools, about a third expressed neutrality on the issue, with comments indicating that their tools were more integrated and did not force them to save their VE every time they made a switch.

P1_{C/AP} Authoring Process

Confusion about the different tools, unnecessary saves or backup and tool incompatibility all contributed negatively towards the authoring process. In the survey, most of the authors who used one tool indicated that they could complete a small VE in a shorter amount of time than those who used multiple tools or tools with multiple modes (see **Error! Reference source not found.**). As only six of those interviewed used a single-tool authoring system, the sample size was too small to verify any statistical significance.

Additionally, there seems to be two main ways in which authors dealt with multiple tool setups, depending on how tightly integrated the different tools were (see Comment P1_{C2}). **Error! Reference source not found.** shows the ratings for the effect

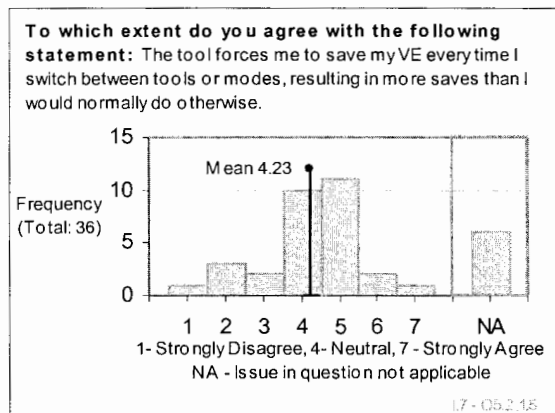


Figure 19: Question 5.2.1.5 from the survey which enquired authors on amount of saves when using multiple tools. (NA indicates users not affected by the issue.)

of using multiple tools on the authoring process, split into how the authors created virtual environments. Furthermore, most of the survey participants affected by this issue rated it as contributing a higher than significant negative effect to the overall authoring time as shown in **Error! Reference source not found..**

Sample Comments:

'Using multiple tools allow [sic] me to focus on each individual task on its own, but I have experienced some incompatibility problems at times' – author who indicated that he created the environment and specified interaction separately (He gave a score of 5 to question 5.3.1.2).

'I get very frustrated having to switch between tools all the time' – author who indicated that he switched between environment building and interaction specification frequently and scored 7 for question 5.3.1.2.

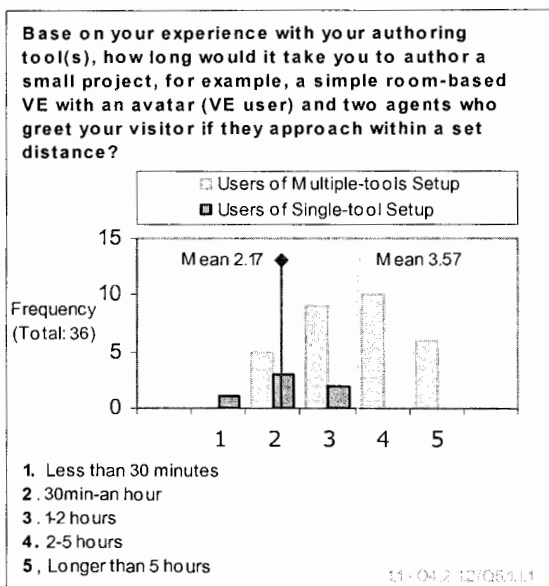


Figure 22: Amount of time authors think it would take them to author a small VE, grouped into those using multiple tools and those that do not. (Results from PS Survey Question 4.2.1.2, cross-referenced with Question 5.1.1.1)

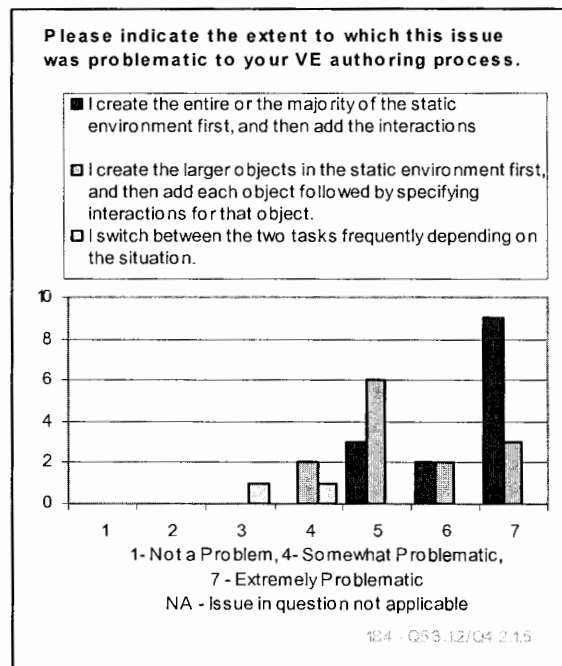


Figure 21: Question 5.3.1.2 from the survey which enquired authors on the effect of using multiple tools on their authoring process. (NA indicates users not using multiple tools.)

P1_{C/LP} Learning Process

It seems logical that learning to use multiple tools (or significantly different multiple-modes tool), will take longer than learning how to use a single tool (or tools which share a similar interface so that to the user, it seems like a single tool). This seems to be reflected in the survey results (see Figure 23). Also more than half of those surveyed and interviewed indicated that this issue was more than somewhat problematic while they were learning how to create a VE (see Figure 24).

Sample Comments:

'I only use one tool, but I felt that there are two very different aspects to learning how to use the tool.' – user of a single tool setup who indicated it took him/her 3-6 weeks to master it, and scored a 6 for question 5.2.2.1.

'I started from scratch and had to learn everything from scratch. It took me a really long time, but I think there are areas which were just difficult, and which we weren't taught, like modeling.' – user of a multiple tool set who took longer than 6 weeks to master it, and scored a 7 for question 5.2.2.1.

P1_{CVE} Overall Quality of VE

During the interviews, authors acknowledged that multiple-tool setup often resulted in delays in the authoring process, but it did not usually badly affect the quality of the resultant virtual environment. Most negative effects involved becoming frustrated with the tedious switching between tools, often with saving to different files between them (for back-up purposes). Some even stopped authoring knowing that more could be done to improve the virtual environment.

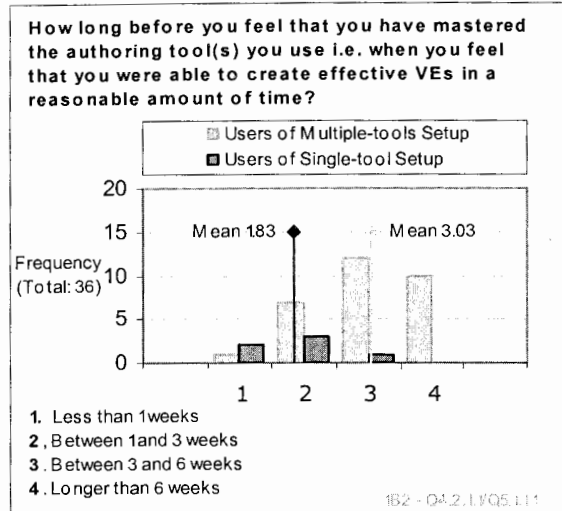


Figure 23: Amount of time authors took to master their tool(s), grouped into those using multiple tools and those that do not. (Results from PS Survey Question 4.2.1.1, cross-referenced with Question 5.1.1.1)

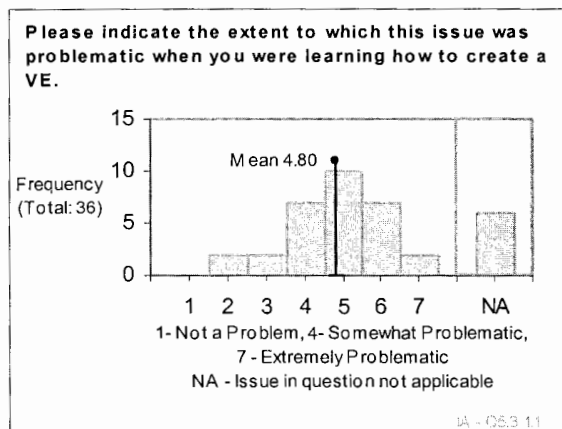


Figure 24: Question 5.2.2.1 from the survey which queried authors on the learning aspect of using multiple tools. (NA indicates users not affected by the issue.)

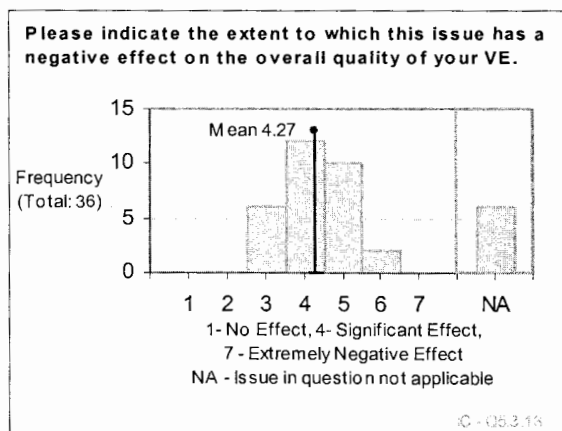


Figure 25: Question 4.2.2.3 from the survey, which queried authors on the effect of using multiple tools on the quality of their VEs. (NA indicates users not using multiple tools.)

5.2 Compile and View Model (P2)

Often authoring tools require all physical environment creation to be completed, followed by a compilation step, before the end-product can be reviewed. For many tools, this is the case when specifying interaction, while for some; a long step is needed to do complex calculations for light or physics subsystems. In some ways, this is similar to programming: an author edits code using an interface which reflects the end-result in one way and, after compilation, displays the result in another. This approach has been found to be problematic in other non-VE related interfaces [42][45]. Users find it frustrating when a resulting product is unsatisfactory, as they are required to enter back into edit mode to make the necessary updates. Non-VE related interfaces with this problem include the older generation of word processors and generally tools with non-WYSIWYG interfaces.

In the initial interviews, all authors used tools that require an intermediate step before the actual end results can be seen. In the survey, 83% (30 out of 36) of the participants also used similar tools. It is interesting to note that there is almost an identical overlap between those not using multiple tools and those not using compile/view tools. This suggests that single tool packages are almost all WYSIWYG packages.

5.2.1 Comments

P2_{C1} Different Representation

Often the intermediate representation displayed during the editing phase is completely different to the actual end-product after *compilation*. An extreme example is when the editing platform is completely different to the one that is used to run the environment, such as with a head-mounted display. There are many consequences of this.

Authors had to mentally translate between the different representations. This cognitive overhead usually resulted in the process taking longer than is necessary. Participants in the survey also concurred with this point (see Figure 26).

In the worst instance, it could lead to an inaccurate environment, as authors grew tired of misrepresentation and/or genuinely did not know how the mapping between the two worked. In the survey, authors rated the compile

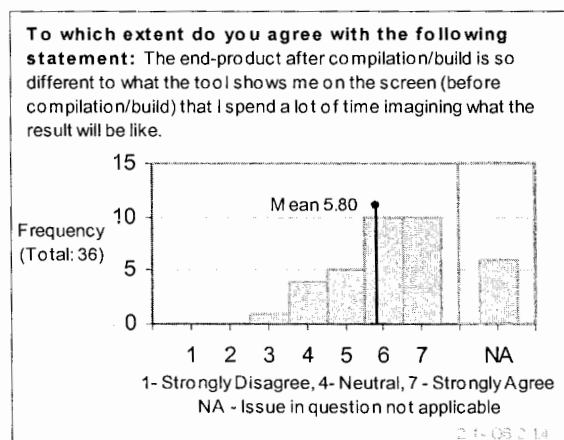


Figure 26: Question 6.2.1.4 from the survey, which queried authors on the difference in representation between edit and view modes. (NA indicates users not affected by this issue.)

execute model as having a very negative effect on the quality of the resultant VE (see Figure 27).

In any case, authors eventually had to learn the mapping between editing representation and the actual end-results. As this was not often documented or otherwise made known to the user in detail, they often had to figure this out by themselves. This was mostly achieved by trial and error and, as such, this skill could take a long time to develop. Participants in the survey also indicated that this issue significantly affected the overall quality of their VEs (see comment P2_{C/VE}).

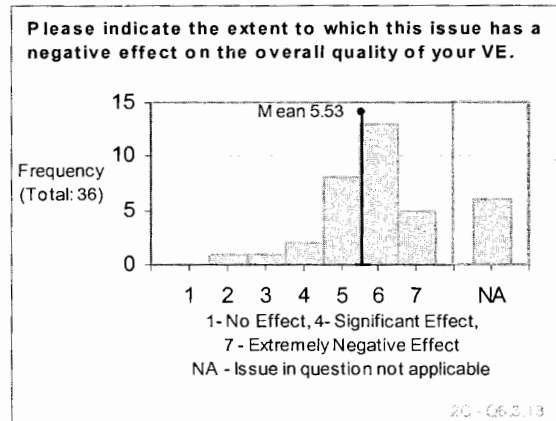


Figure 27: Question 6.3.1.3 from the survey, which queried authors on the effect of using compile/view tools on the quality of their VEs. (NA indicates users not using compile/view tools.)

P2_{C2}: Lack of Immediate Feedback

Since the result could not be seen immediately after the author had performed the editing, the problem lay chiefly in the lack of direct feedback during the editing phase which can results in two scenarios:

- The author could choose to perform numerous small edit-and-build steps to be able to continuously see the end-product. This method was time consuming but often resulted in a product closer to the author's expectations. Authors were also able to realise mistakes at an earlier stage and could correct them quickly and easily (see Figure 28). 28% of the participants indicated that they fell into this category.

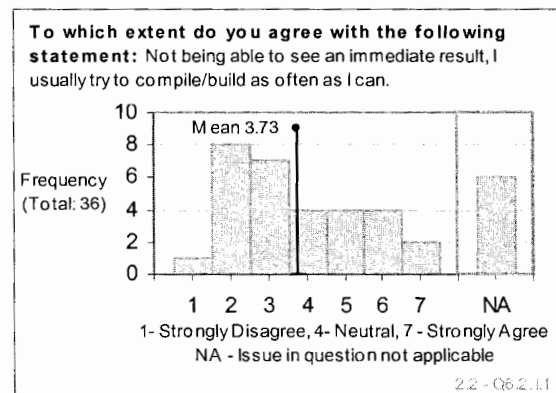


Figure 28: Question 6.2.1.1 from the survey, which queried authors on the compile/build frequency while using compile-view tools. (NA in indicates users not using compile-view tools.)

- With some tools the build or compilation process was lengthy and this had the opposite effect of encouraging the author to delay the build or compilation process until a number of changes had been made. Although authors could save time by not having to perform the lengthy build or compilation process, more time was often wasted when the end-product is not what was expected. In such cases, the author would need to retrace all the steps and try to locate where the mistakes occurred (see Figure 29). 42% of the participants fell into this category.

P2_{C3} Build-all-then-view Mentality

In compile and view type systems, authors kept the compilation frequency to a minimum, often thinking of the process as a set of long build steps followed by a compilation step. This is reflected in the result of the survey (see Figure 30).

P2_{C4} Different Control for Edit and View Modes

One would expect there to be significant differences in interfaces of the edit and view modes (or tools). For a lot of users this is the case as shown in Figure 31.

Most of the comments concerned the difference in controlling the camera in the two modes.

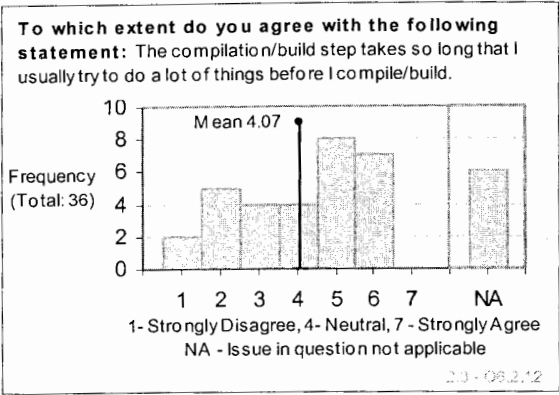


Figure 29: Question 6.2.1.2 from the survey, which queried authors on amount of work done in between compile/build steps while using compile-view tools. (NA indicates users not using compile-view tools.)

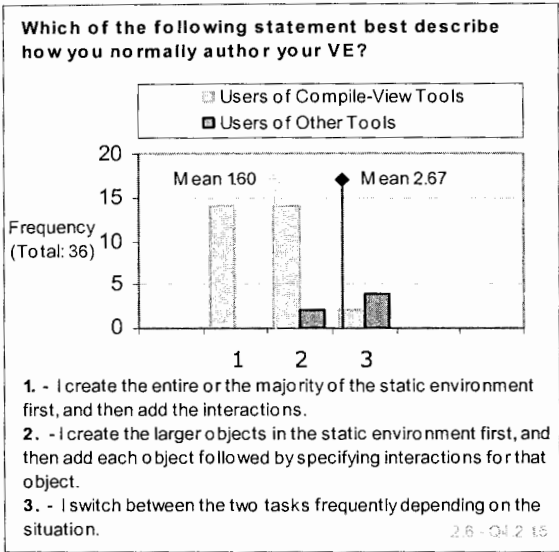


Figure 30: Amount of time authors took to master their tool(s), grouped into those using multiple tools and those that do not. (Results from PS Survey Question 4.2.1.1, cross-referenced with Question 5.1.1.1)

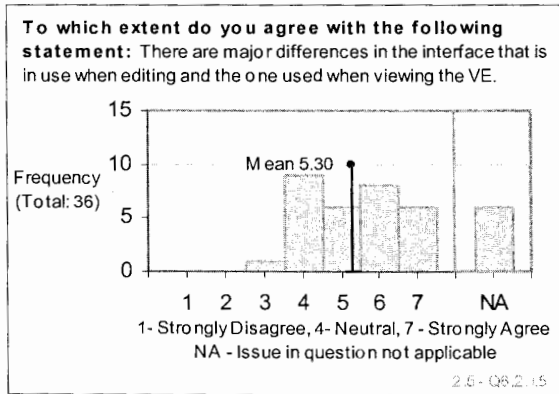


Figure 31: Question 6.2.1.5 from the survey, which queried authors about interface differences in their compile-view tools. (NA indicates users not using compile-view tools.)

P2_{C/AP} Authoring Process

Difference in representations between the pre-compiled and the actual results were said to have made the authoring process difficult by a lot of authors. As one author commented, the difference effectively means that they have to 'guess their way' through the entire process. Additional comments also generally attributed the compile and view method as one of the factors that unnecessarily lengthened the authoring time. There was a trend that users of compile-view tools indicated longer times for creating a simple VE (see Figure 33), but we were unable to verify or rebuke this trend statistically because of the number of samples involved.

Sample Comment: *'Perhaps the worst part of the tools that I used – the waiting time is not too much, but the result is sometimes not as expected'* – User of compile-view tool who scored 6 on the effect rating.

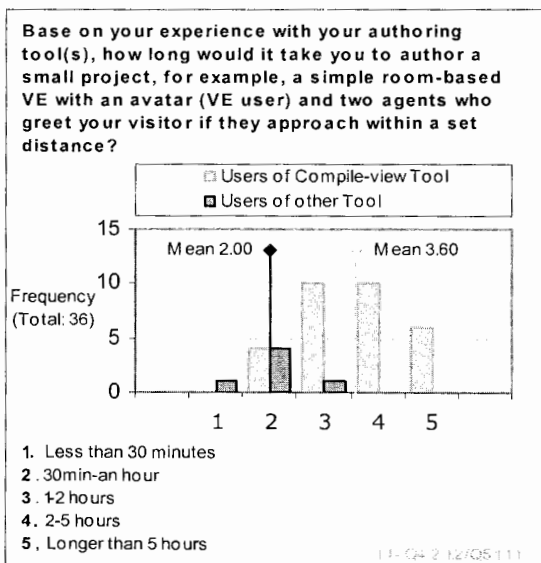


Figure 33: Amount of time authors took to master their tool(s), grouped into those using compile/view tools and those that do not. (Results from PS Survey Question 4.2.1.1, cross-referenced with Question 5.1.1.1)

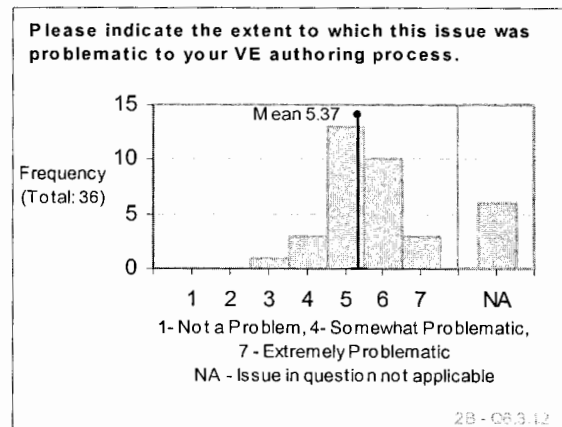


Figure 32: Question 6.3.1.2 from the survey, which queried authors on the effect of using compile and view tools on the authoring process (NA means authors not using compile& view tools)

P2_{C/LP} Learning Process

Often the difference in representations between the pre-compiled and actual end-product could only be learnt through experience. There was often no quick way to do this and this probably led a few authors into commenting that it took a long time for them to learn what is to be expected. The interface differences in the two modes could also play a part in the learning process. Figure 34 illustrates how authors rated this issue's effect on their learning process.

Sample Comment: *'It's bad that nothing matches what I see before the build, even the keys are different.'*

P2_{C/VE} Overall Quality of VE

The majority of authors who took part in the survey indicated that the compile-view model had a significant negative effect on the quality of their virtual environment (see Figure 35). A common problem was that the frustration experienced by authors often led them to be satisfied with a lower quality of VE than would otherwise be possible. The frustration could be attributed to the unexpected nature of the result.

Sample Comment: *'I have a lot of problem resizing object to the right size, it's always the wrong size in the actual thing.'*

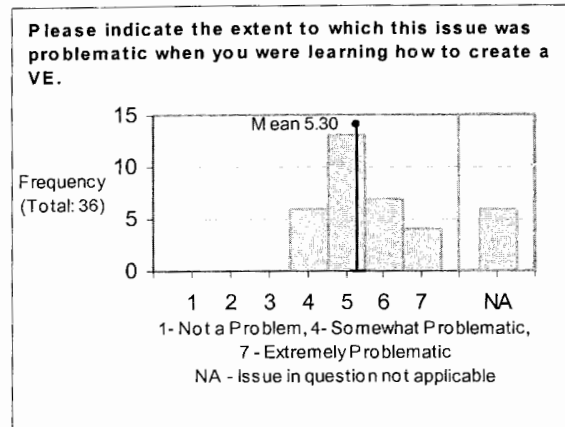


Figure 34: Question 6.3.1.1 from the survey, which queried authors on the effect of using compile and view tools on the learning process (NA means authors not using compile& view tools)

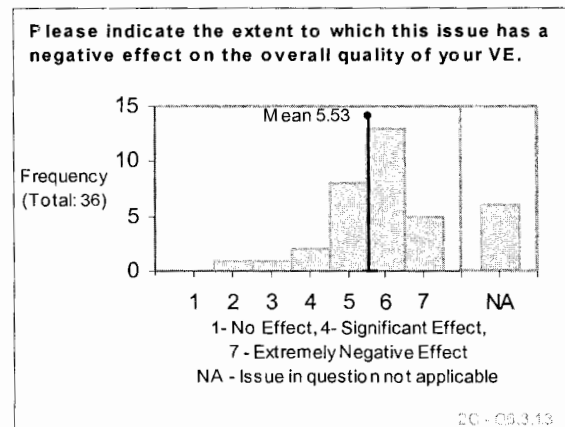


Figure 35: Question 6.3.1.3 from the survey, which queried authors on the effect of using compile& view tools on the quality of their VEs. (NA indicates users not using compile& view tool.)

5.2.2 Analysis

There are typically three reasons why creators of authoring tools may have resorted to the compile-and-execute model.

P2_{A1} Difference in Platforms

The first reason applies to systems where there is a significant difference between the editing and the target platforms. The difference could be partially in hardware levels such as processing power, memory or the presence of a graphics accelerator. The actual hardware platform might be totally different such as when authoring for a head-mounted display on a desktop machine. The difference could also be in software, where the target system might run on a different operating system. This means that before the VE could be viewed, it has to be compiled for the targeted system first.

P2_{A2} Difference in Interfaces

This is similar to the previous reason. In some cases, the authoring interface and the viewing interfaces were different. For example, clicking objects in an authoring interface selects it, while in the viewing interface, an action is performed. This means that there is the need to interchange between the two interfaces.

P2_{A3} No Real-time update

This reason is due to the reliance on a compiled programming language for specifying interactions. In some tools, objects in the VE and the code governing the interaction cannot be linked in real time due to various reasons including complexity and efficiency.

P2_{A4} Time-frame Issues

The next reason involves timing issues. VEs often run in their own time frame (as many interactions are time-based). This means that many VEs have a distinct beginning; the point where the VE must start from every time.

5.3 Mixed 2D/3D Interfaces (P3)

Most VE authoring environment tools are predominantly desktop-based, even when creating a VE for other platforms. Being based on a two-dimensional display, authoring tools rely heavily on existing two-dimensional interfaces provided by the underlying system. Typical tools provide a viewing point where the three-dimensional VE can be seen, surrounded by two-dimensional widgets such as a menu bar, properties table, feedback pane and perhaps a programming pane where interactions can be specified. A very popular method of object placement in VE authoring tool is the two-dimensional three viewpoints interface. This is looked at further as an issue on its own in the next section. Figure 40 shows screen captures of tools that contain different levels of the two- and three-dimensional interfaces.

All authors in both the interviews and the surveys indicated that they used tools that have mixtures of both 2D and 3D interfaces.

5.3.1 Comments

P3C1 Pointer Confusion

Different interfaces employed different means of pointing. While both may use a cursor with appropriate number of dimensions, the transitions between them were not smooth. The cursor often disappeared and reappeared in a different position on moving from one type of interface to another. Some tools employed different method to control the two cursors; often with additional cursor controls for movement along the extra dimension in the VE pane. This contributed negatively to the authoring process as it was often frustrating to the author as well as causing delays (See Figure 36).

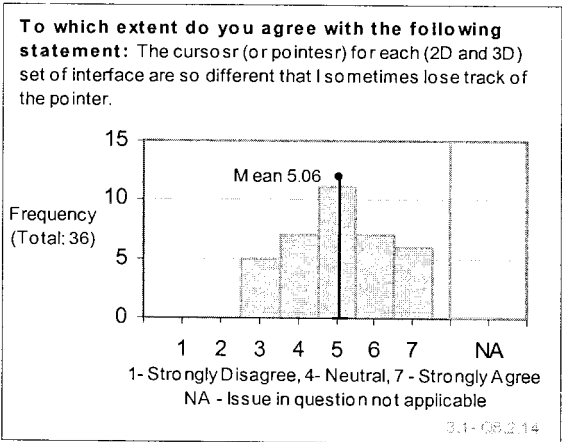


Figure 36: Question 8.2.2.4 from the survey, which queried authors on the difference in pointers in 2D and 3D portion of their mixed interface (NA represents users for whom there are no apparent separation between 2D and 3D)

P3_{C2} Constant switching between mindsets

Authors were forced to change mindset between the flat (two-dimensional) nature of the interface and the three-dimension one previewing the VE. For example, in a two-dimensional area of an authoring tool, moving the mouse moves the cursor as per usual, but in three-dimensional tools, moving the mouse may move the camera and not the cursor. Furthermore, for some tools, the flat widget does not closely relate well to its three-dimensional counterpart, requiring authors to have to learn this extra knowledge. For example, when editing properties of an object through the properties pane, the authors often felt that the flat widget and the three-dimensional object were not linked. Furthermore, the two types of display could reinforce other kinds of edit and view modes outlined in the previous section. Consequently, authoring times could be effected negatively. This could also lengthen initial learning time as authors have to form mental mappings between the two-dimensional widget and its effect in the VE (see Figure 37).

P3_{C3} Cluttered interface

The mixture of two- and three-dimensional interfaces often resulted in a cluttered work area. In most authoring tools, the interface was even made more cluttered by the inclusion of many buttons and fields which, at one time or another were non-functional. Many authors in the survey agreed that this is the case (see Figure 38).

Many authors also commented that they were also confused by the cluttered interface as this makes the tools seem overly complex. It made the correct button or pane harder to find, even if the

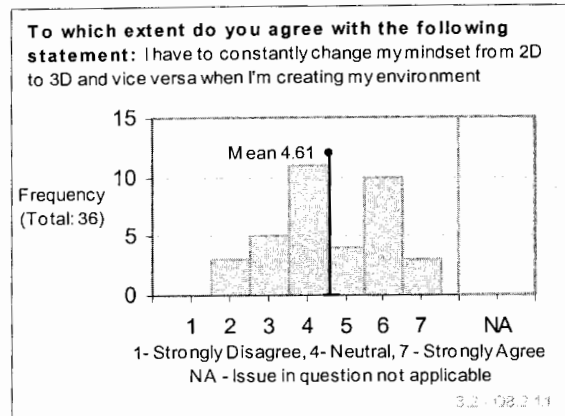


Figure 37: Question 8.2.2.1 from the survey, which queried authors on mindset change when authoring in 2D and 3D portions of their tools (NA indicates users for whom there are no separation between 2D and 3D-based interfaces)

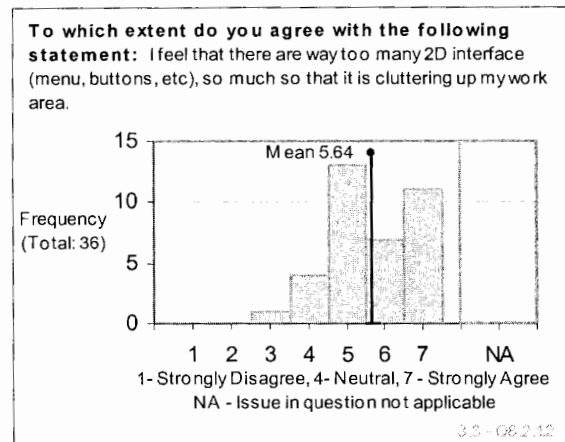


Figure 38: Question 8.2.2.2 from the survey, which queried authors about 2D interface clutter on their tools (NA indicates users for whom there is no separation between 2D and 3D-based interfaces)

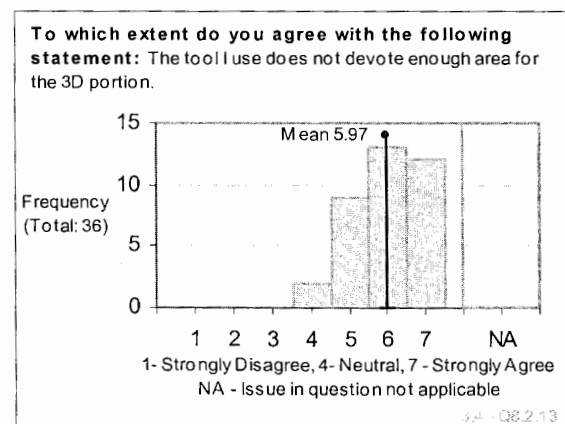


Figure 39: Question 8.2.2.3 from the survey, which queried authors the amount of view area dedicated for the 3D portion of their tools. (NA indicates users for whom there is no separation between 2D and 3D-based interfaces)

authors knew exactly which one to press or enter text into. Figure 40 shows how cluttered the interface can be.

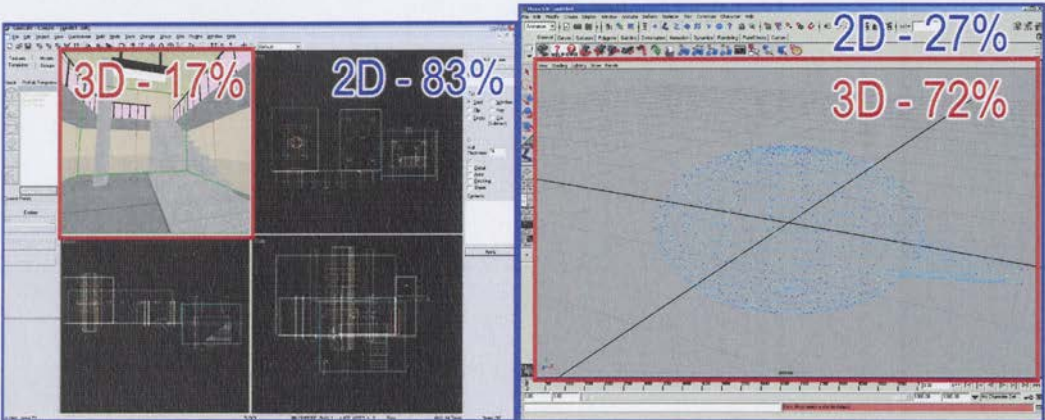


Figure 40: Screen captures from GenEdit (left) and Maya (right) in single pane mode. Each shows the relative default sizes of 2D and 3D aspects of the interface.

P3C4 Reduced 3D viewing area

Some authoring tools had a reduced VE viewing area. This often led to many problems including inaccurate VE placement as the view shown in the preview pane is not seen in the same proportion and size by the VE visitor. Authors also experienced difficulty in selecting and placing objects with the reduced view. Figure 40 also shows the relative sizes of two- and three-dimensional areas of the interface. Most authors who took part in the survey also indicated that the tools they use do not devote enough area to show the 3D scene (see Figure 39).

P3C5 Delayed Update

A number of authors have commented that the lag between editing in the 2D interface and the update in the 3D environment could lead to frustration. Specific examples include changing values of attributes in the 2D interface portion and having the changes reflected a few seconds after entering the 3D portion. More than half of those participated in the survey indicated that this problem also affects them to some degree (see Figure 41).

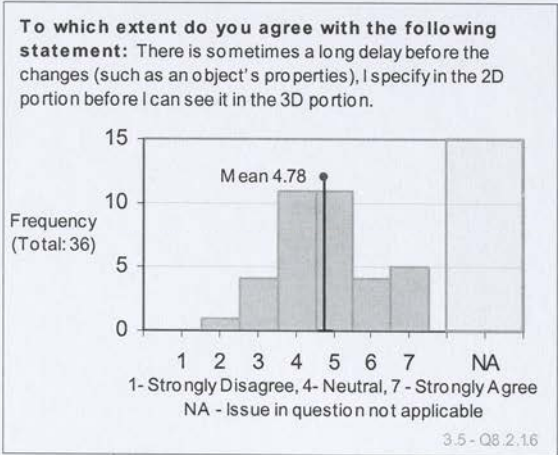


Figure 41: Question 8.2.2.3 from the survey, which queried authors about the amount of view area dedicated for the 3D portion of their tools. (NA indicates users for whom there is no separation between 2D and 3D-based interfaces)

P3_{C6} Build-all-then-view Mentality

Most authors saw the 2D portion of the interface as the 'editing section' where most of the environment manipulation should take place (see Figure 42). Some saw it as a window to the VE. Most authors also thought of the 3D portion as the preview window for the VE. The perceived separation can further reinforce the *build-all-then-view* mentality in authoring a VE.

P3_{C7} Difficulties in directly editing an object's attributes

In the survey, more than half the author rated changing an object's physical properties as being of more than average difficulty (see Figure 43). The mixture of 2D and 3D interfaces may have contributed to that, but since every authors who took part used tools which have this mixture, we were not able to get a feel as to whether this is the case.

P3_{C/AP} Authoring Process

It was apparent from the comments that authors initially were not concerned with this issue. However, as they considered the issue, new revelations were realised. Comment made on this issue varied from *'I could see that it would be a problem, but it hasn't really occurred to me'* to *'It takes me a long time to find out where my cursor is after I move it.'* Missing cursors seemed to cause the most problem, judging from the comments. Figure 44 (top) illustrates the survey result.

Other negative contributing factors to the authoring process included cluttering of the interface and the amount of viewable area of the different types of interfaces.

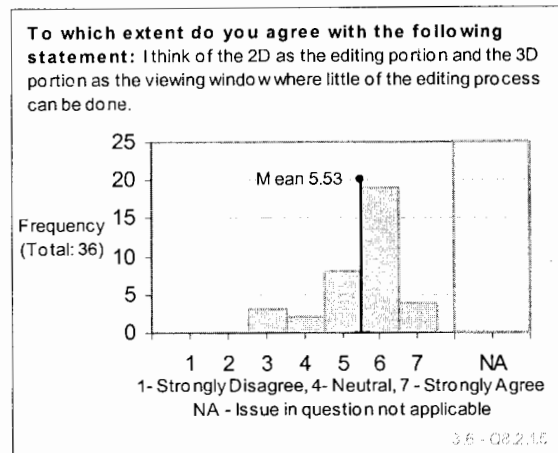


Figure 42: Question 8.2.2.5 from the survey, which queried authors on how they see the difference in 2D/3D portions of the interface. (NA indicates users for whom there is no separation between 2D and 3D-based interfaces)

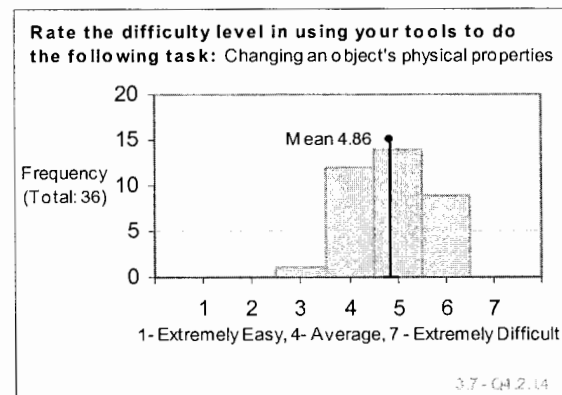


Figure 43: Question 8.2.1.4 from the survey, which enquired authors on the difficulty of changing physical properties of an object in the virtual environment.

P3_{C/LP} Learning Process

In the survey, authors gave the effect on the learning process a mid-scale rating of *somewhat problematic*. Once again, a lot of comments pertained to the cursor appearances; with a few complaining about their 2D appearances in the 3D VE or the inconsistency between the two.

P3_{C/VE} Overall Quality of the Virtual Environment

The fact that authors did not think of this issue as a problem initially could be taken as an indication of how well authors have overcome this issue, possibly reflecting on the rating they gave on the effects of the overall VE quality. They have learnt to deal with it effectively; in fact, there were no comment regarding how they might have to compromise the quality because of this issue.

5.3.2 Analysis

P3_{A1} Extension of existing interfaces

This mixture of two- and three-dimensional interfaces seems to have been the result of extension of existing two-dimensional interfaces. Most tools have a set-up which is similar to visual programming tools such as Delphi or Visual Studio. In a way it was a natural choice; the two authoring processes (VE and Software) share many common traits. Additionally, VE building was not the first occurrence of dealing with the representation of three-dimensional objects on two-dimensional plane. Architects and engineers have been doing this for a long time and VE authoring tools have borrowed their proven representation methods and use them in their tool. An example of this is the three-viewpoint placement interface.

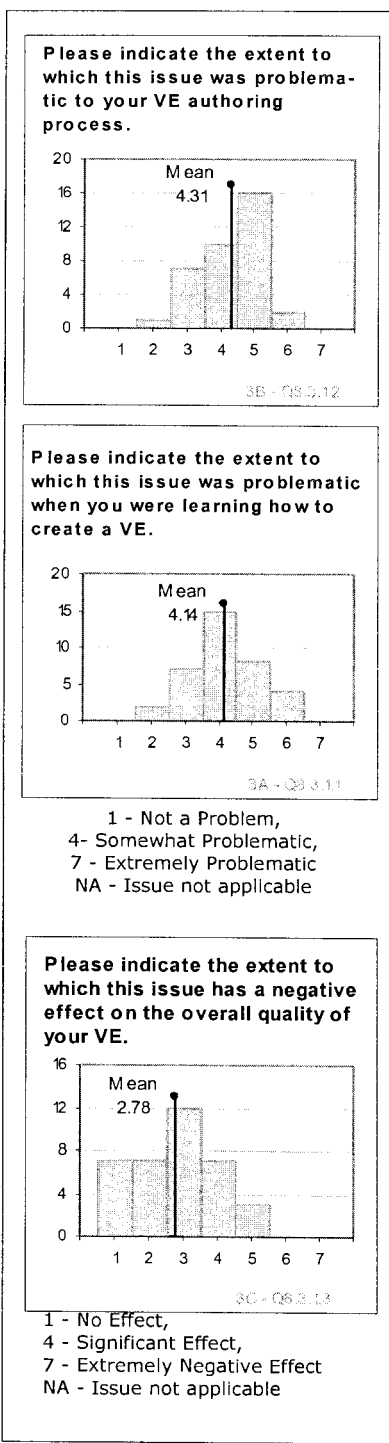


Figure 44: Results from survey Questions 8.3.1.1-8.3.1.3 which queried the users on the effects on learning, authoring and quality of VE, by the 2D/3D mixture of user interfaces in the tools that they used.

5.4 Multiple Viewpoints Object Placement Interface (P4)

It was discovered that most VR authoring tools employ by participants featured multiple pane interfaces for some functionalities. Each pane showed different viewpoints. Some also employ the equivalent but with tabs instead of panes. This type of interface has been primarily used in modelling tools such as Maya or Blender, but has recently been used in environment building tools as well. Typically there is a pane that allows the author to choose any of the available perspective views for reviewing purpose, and another three panes showing different viewpoints. The latter three are the orthographic viewpoints used in *engineering drawing* with top-down, side-on and front views. The interface, like engineering drawings accurately represents the three-dimensional environment on a two-dimensional screen. Examples include GenEdit (see Figure 40) – a VE authoring tools for creating static environment. Maya, though a modeling tool, is used by many authors to build static environment. Its default interface also contains three viewpoints; see Figure 45.

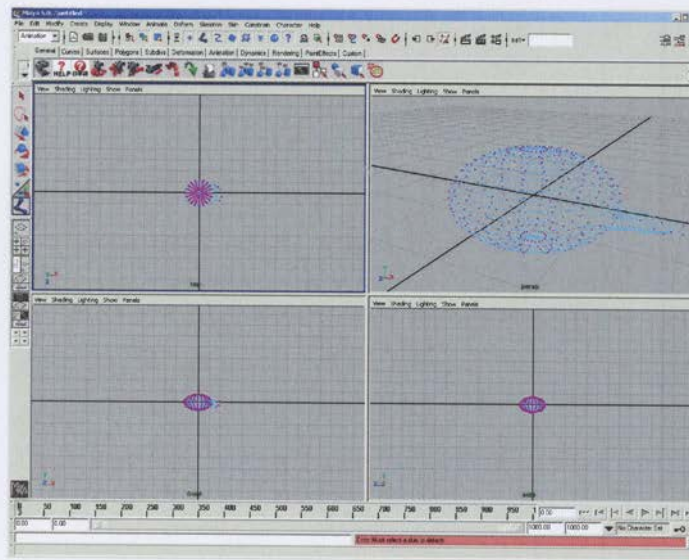


Figure 45: Maya's default interface has four sub-panes, each showing a different viewpoint – the user may change to a single pane mode (see Figure 40).

5.4.1 Comments

P4_{C1} Confusing interface

Authors who were not familiar with engineering drawings found the interface very confusing to learn and use. Even authors who had been using it for over a year have difficulties using it. Although the three-viewpoint system works well for a static image, it was not so easy to use dynamically to place or move objects. This was mainly because authors had to shift focus to different panes for movement along different axes. In some tools, movement of the mouse and/or the keyboard affects

movement in all planes. This often led to author unintentionally moving an object along one plane while intending to move it along another. More than half the authors find the multiple viewpoints/pane interface confusing and complex (see Figure 46).

P4_{C2} Cluttering Interface

A higher number of people thought that the interface wastes a lot of space and could have been better used for something else (see Figure 47). In the interview, several authors indicated that it might be better to only call up the other viewpoints when needed, but this might effectively *hide* the interface, making the initial learning curve higher. Although most authors interviewed thought that the interface afforded them the ability to be precise in the placement of objects, most authors in the survey indicated that they did not need this feature all the time (see Figure 48). Some of the comments also suggested that the feature be asked for and not be present by default. It should be noted that there are tools that can do this but some authors were not aware of this feature.

P4_{C3} Difficulty in moving, placing and locating objects

Ironically, systems designed for accurate placement of objects often resulted in placement error. Some authors found it extremely difficult to use to place objects accurately and often ended up with compromised results. The majority of authors interviewed and surveyed agreed that the reason for this is due to the nature of the placement interface. This problem is often compounded with the absence of gravity and/or collision detection, leading to objects going

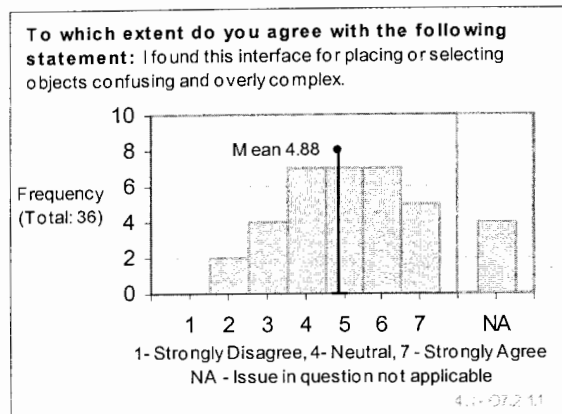


Figure 46: Question 7.2.2.1 from the survey, which queried authors on the complexity of placing or selecting objects using the multiple-pane placement interface. (NA indicates authors not using this type of interface)

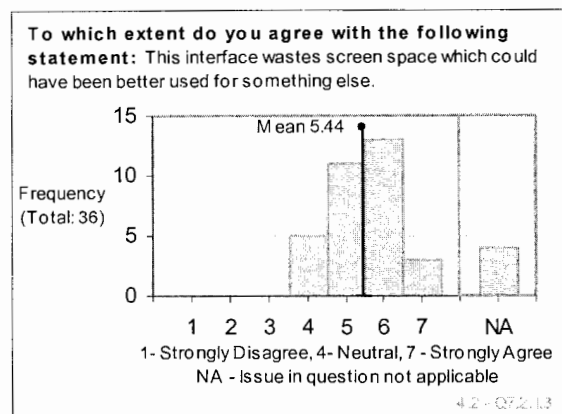


Figure 47: Question 7.2.2.3 from the survey, which queried authors on the complexity of placing or selecting objects using the multiple-pane placement interface. (NA indicates authors not using this type of interface)

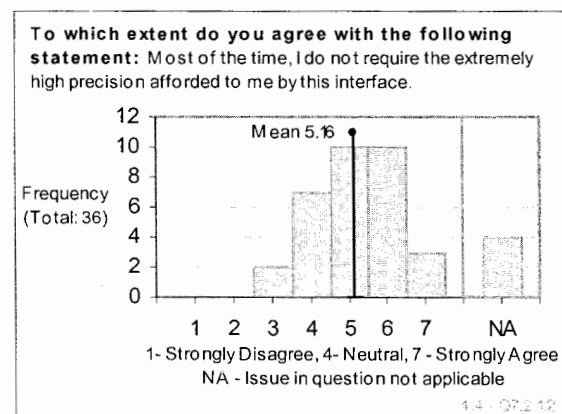


Figure 48: Question 7.2.2.2 from the survey, which queried authors on the complexity of placing or selecting objects using the multiple-pane placement interface. (NA indicates authors not using this type of interface)

through walls, floors or other objects unintentionally.

Another interesting observation is that all of the users of non-multiple pane interface rated the task of moving an object as average to lower than average difficulty (see Figure 49). However, with such a small number (11% of those surveyed), this was not conclusive.

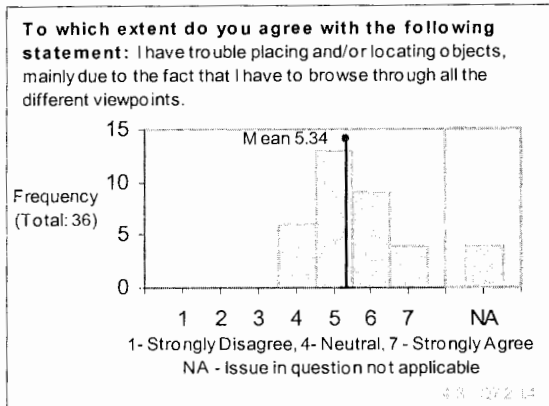


Figure 50: Question 7.2.2.2 from the survey, which queried authors on the complexity of placing or selecting objects using the multiple-pane placement interface. (NA indicates authors not using this type of interface)

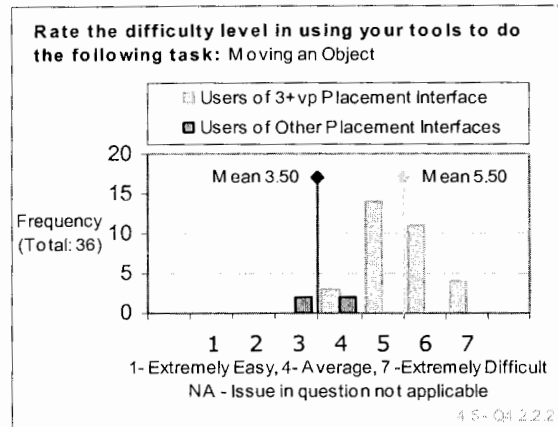


Figure 49: Difficulty rating for the task of moving an object while editing, grouped into those using multiple pane/viewpoint placement interface and those that do not. (Results from PS Survey Question 4.2.2.2, cross-referenced with Question 7.1.1.1)

P4_{C/AP} Authoring Process

The confusing nature of the multiple pane placement interface was one of the most commented on contributors to the authoring process. Authors did not seem to be able to master this interface easily. Comments that were made included *'Can't get my head around which windows means which view ... just have to test as I go along.'* And *'Being an engineer, I find the interface simple enough.'*

In the survey, all of the authors rated the effect to the authoring process as higher than *'somewhat problematic'* level (see Figure 51 top)

P4_{C/LP} Learning Process

This is the only issue which most authors have indicated that they had not got right, even after years of experience using the tool. This would suggest that authors struggle to learn how to use this interface effectively. Survey results seemed to also confirm this with most authors indicating a higher range of the scale as seen in Figure 51 (middle).

P4_{C/VE} Overall Quality of Virtual Environment

It was clear from both the interview and the survey that this issue negatively affected both the learning and authoring processes. However, authors did not rate the issue as one of the more influential factors affecting the overall VE quality (See Figure 51 bottom). Along with some comments, this would indicate that authors feel that even though they find the interface difficult to master and use, they had found a way around it that does not sacrifice the quality of the VE. There were some authors, however, who commented that they had to settle for a less than satisfying end-product, but because most of the negative effect is on the position of objects, this was not a major problem.

5.4.2 Analysis

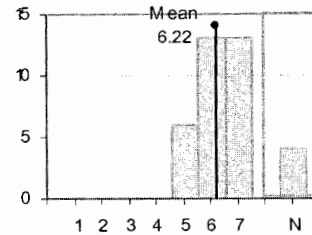
P4_{A1} Mixture of Interfaces

Part of this problem is basically a specific example of a mixed two- and three-dimensional interface discussed in 5.3 Mixed 2D/3D Interfaces (P3). The placement system was developed from an already proven engineering drawing scheme. It provides extremely accurate placement of objects which other interfaces might find hard to match.

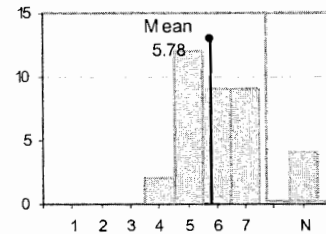
P4_{A2} Lack of gravity and collision detection

The majority of complaints concerning the difficulties in using this interface involved the lack of gravity and/or collision detection. Objects often go through walls, floors and other objects. This problem is compounded when authors have to make sure that this does not happen by monitoring all three viewpoints. On several occasions, an attempt to correct an object's placement in one viewpoint causes it to move out of place in another.

Please indicate the extent to which this issue was problematic to your VE authoring process.

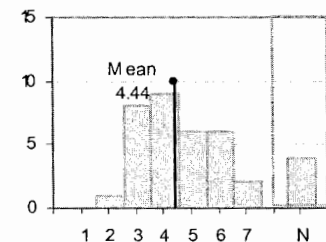


Please indicate the extent to which this issue was problematic when you were learning how to create a VE.



1 - Not a Problem,
4 - Somewhat Problematic,
7 - Extremely Problematic
NA - Issue not applicable

Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.



1 - No Effect,
4 - Significant Effect,
7 - Extremely Negative Effect
NA - Issue not applicable

Figure 51: Results from survey Questions 7.3.1.1-7.3.1.3 which queried the users on the effects on learning, authoring and quality of VE, by the multiple pane placement interface.

5.5 Selecting and Editing Objects (P5)

Before any existing object could be edited it must be selected. In some authoring tools used by some of the interviewed authors, object selection was only possible with a list-based widget. However, some tools provided a direct selection interface where the author may directly click on the object to select it. This was either done by moving the selector/cursor to touch the required object or by projecting a ray onto the required object. This is also known as the direct selection method. Indirect selection involves user selecting objects via an interface that is indirectly linked to the object. Some tools falling into the latter category forced authors to select an object by locating its name from a drop-down list, or by typing in the name into a text editor.

Once an object was selected, it could be manipulated either directly or indirectly. As an example of direct editing, author can increase the size of an object by dragging a point on the object in a specific way. In tools with indirect manipulation, users might have to change a value in a text box in order to manipulate an object size. In some tool, the text box might be placed next to the object and the changes happen in real time, with the user seemingly directly manipulating that object.

All of the authors interviewed used tools with a mixture of direct and indirect selection and manipulation. Authors reported that most tools let them do direct manipulation [59] for a simple operation such as resizing, rotating or moving, but indirect manipulation for more complicated operations such as changing colours or some complex attributes. The worst case of indirect selection and manipulation was arguably the scripting system that requires a high amount of programming.

In the survey, which focussed specifically on selecting objects for editing or creating interactions, a quarter of the group indicated that the system allowed for direct selection from a virtual environment or the preview screen.

5.5.1 Comments

P5_{C1} Confusing interface

In the interview, authors indicated that the main drawback of indirect selection is that it forced them to have to form mental link between the object in the VE and the indirect representation of them (e.g. Name) in the interface. In some cases this led to confusion. However, this problem could be resolved by using a consistent naming scheme. In the survey, 87% of authors surveyed also agreed that this issue is confusing and

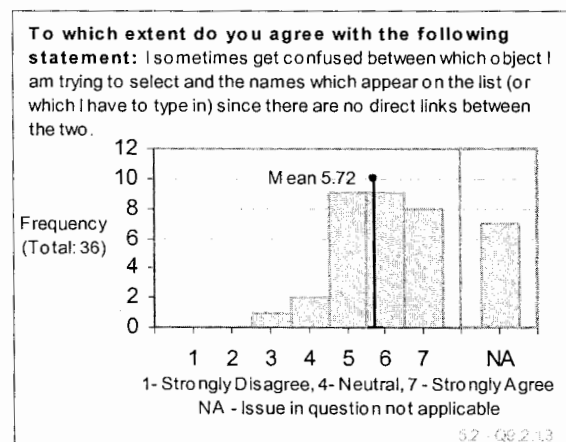


Figure 52: Question 9.2.2.3 from the survey, which queried authors on the level of confusion related to indirect selection. (NA indicates authors not using this type of interface)

that it contributed to the lack of direct links between the two representations (See Figure 52).

P5_{C2} Difficult to use

Approximately 60% of authors who were exposed to indirect selection in their tools agreed that they have difficulty in using the interface (see Figure 53). Almost all of these authors ranked themselves at least as *moderately experienced* in VE creation. Figure 54 shows the difficulty rating for finding and selecting objects and illustrates an apparent trend that authors considered direct selection to be easier than the indirect method.

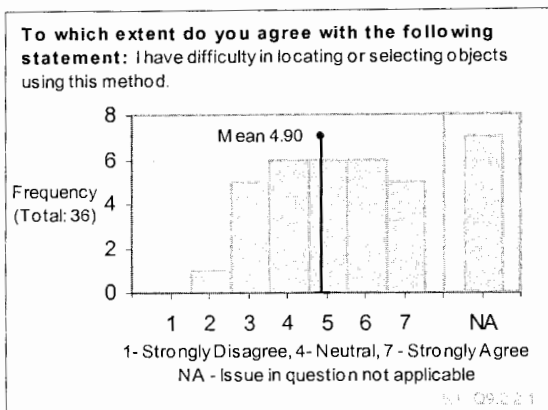


Figure 53: Question 9.2.2.1 from the survey, which queried authors on the level of difficulty related to indirect selection. (NA indicates authors not using this type of interface)

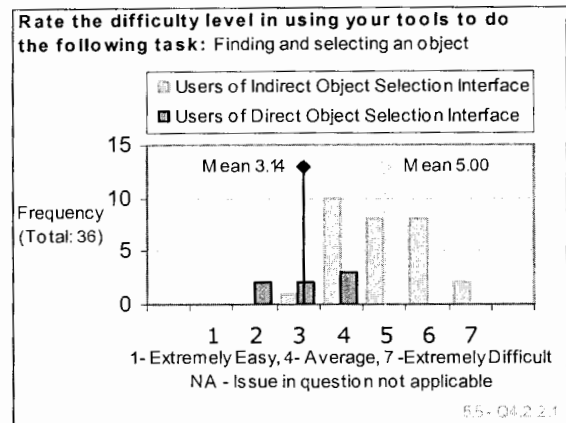


Figure 54: Difficulty rating for the task of finding and selecting an object while editing, grouped into those using direct and indirect methods. (Results from PS Survey Question 4.2.2.2, cross-referenced with Question 9.1.1.1)

P5_{C3} Keeping track of selected object

A consequence of using indirect selection was that authors had to keep in mind the object being edited since the interface for selection was not directly linked to the object. Most displayed an intermediate link to the object, such as the object's name. During the interviews, authors with indirect interfaces indicated that most had difficulty keeping track of the object being manipulated. Those that have the most difficulty seem to be the ones with interfaces that did not focus on, or showed, the picture of the current object. Survey participants also indicated that they also experienced some level of difficulty (see Figure 55).

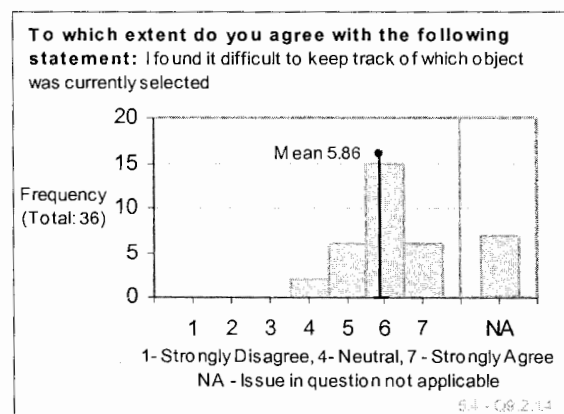


Figure 55: Question 9.2.2.4 from the survey, which queried authors on the level of difficulty in keeping track of current object using indirect selection. (NA indicates authors not using this type of interface)

P5_{C/AT} Authoring Process

Interviewed authors reported that this issue significantly affected their authoring time. In the survey, more than 70% of those using tools with indirect selection indicated that finding and selecting objects takes a long time (see Figure 56). Furthermore, most participants who *strongly agree* used tools where objects' names must be typed in to manipulate them.

Survey participants also indicated overwhelmingly that this issue was problematic to their authoring process (see Figure 57 middle).

Sample Comments:

"There are usually a few hundred objects in my program. I can't remember them all."

"My app provides a list of objects to choose from but with so many I can't tell which is which, perhaps a photo would help?"

P5_{C/LT} Learning Time

Surveyed authors all reported that this issue contributed negatively to their learning process, with 90% reported higher than significant effects (see Figure 57 left). The fact that authors also reported high negative effects for their authoring process indicated that negative effects did not decrease with experience.

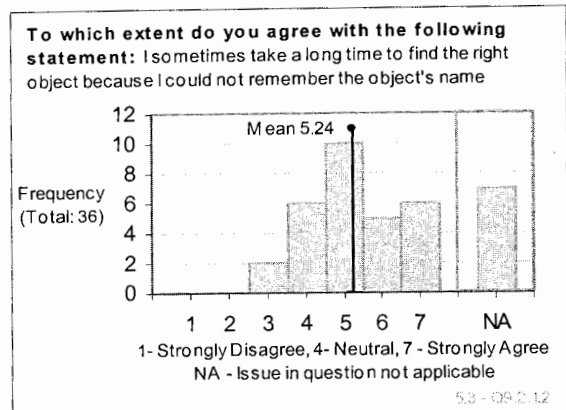


Figure 56: Question 9.2.2.2 from the survey, which queried authors on time taken to find objects using indirect selection. (NA indicates authors not using this type of interface)

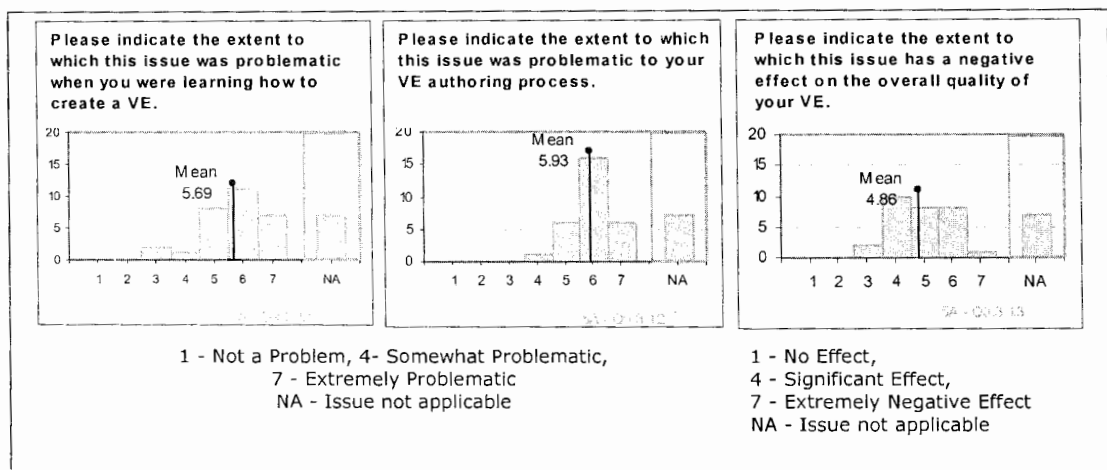


Figure 57: Results from survey Questions 9.3.1.1-9.3.1.3 which enquired the users on the effects on learning, authoring and quality of VE, by the direct and indirect object selection interface in the tools that they used.

P5_{C/VE} Overall Quality of VE

Concerning overall quality of their VE, the issue was felt to have significant effect (see Figure 57 right), but a few of the comments were more related to interaction specification than selecting objects.

5.5.2 Analysis

P5_{A1} Adaptation of existing interfaces

There were usually two scenarios where authors needed to select an object. They could select the object in order to change its attributes or to specify interactions.

Most of the authors interviewed indicated that their tool's attribute changing interface allowed direct selection. This was probably a derivation of the modeling tool interface that allowed users to directly change an object's size by dragging points on the object's mesh. Another method often employed by VE authoring tools is the select-from-a-list approach, which is found in a lot of drawing program like Adobe Photoshop.

Interaction specification, often just an integrated development environment (IDE), uses indirect selection involving users typing in the name of the desired objects. Like advanced IDEs, some employed an auto-complete function to help authors identify desired objects.

5.6 Lack of Real World Mappings (P6)

Most interviewed authors created their VE to mimic a real environment, at least in some aspects. It would then be advantageous for authoring tools to take advantage of these facts and try to mimic real world conditions during the creation process. A lot of problems and complaints experienced by authors during the interview often involved inconsistency mapping between the real world and the simulated environment (or previewed environment) during the authoring process. Some of the related issues are outlined in this section.

5.6.1 Comments

P6_{C1} Lack of gravity & Collision Detection

Most tools place new objects in mid-air and authors had to manually manipulate their vertical position to make them sit correctly on the floor or another object. Most of the authors who took part in the survey also indicated that they also have to manually manipulate the height of objects (see Figure 58).

It was suggested that it might be desirable to have gravity effect on by default so that objects added automatically sit at the correct height. Furthermore, when placing an object, it is fairly easy to move one object into another if there is no collision detection. Collision detection system usually detects when objects are in contact with each other and prevent them from going into each other. The multiple view-point interface compounded the lack of collision detection during authoring by making it difficult for a lot of authors to determine if an object is passing through another neighbouring objects. 83% of authors surveyed also indicated that they had to manually make sure that objects do not go through walls or other objects when they place an object (see Figure 59).

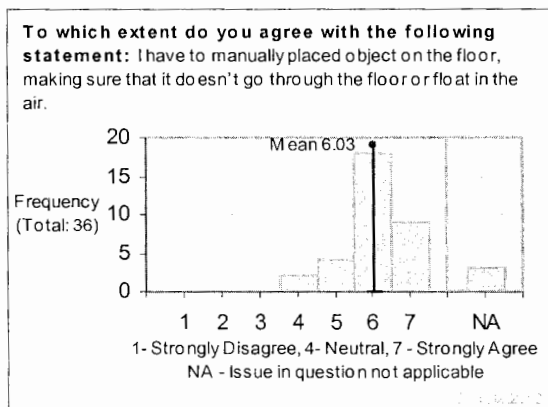


Figure 58: Question 12.2.1.2 from the survey, which queried authors on how they place objects on the floor. (NA indicates not applicable)

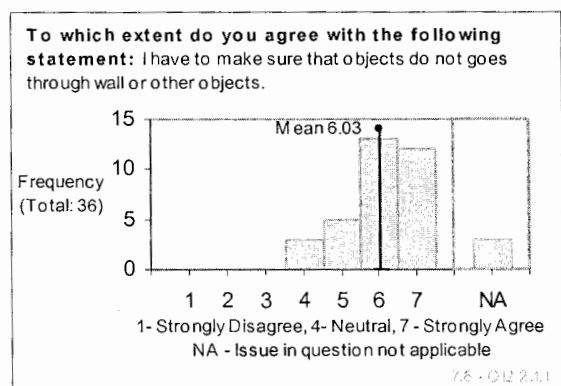


Figure 59: Question 12.2.1.1 from the survey, which queried authors on how they place objects next to walls or other objects. (NA indicates not applicable)

P6C2 Minimal Object Mapping

Objects in VE were often treated by the authoring system as plain vanilla objects, each of them behaving essentially the same as one another. This forced authors to script or create interactions which they considered trivial. For example, a car might have to be scripted to move, a character to walk etc. Many authors would like to see tighter coupling between interactions and the physical appearance of an object.

All but one (97%) of the surveyed authors found that this situation was applicable to them, with 80% of that group agreeing that they had to repeatedly define similar/same behaviours for objects that were similar in nature (see Figure 61). 91% of the applicable group also felt that some objects should have already been defined (see Figure 60).

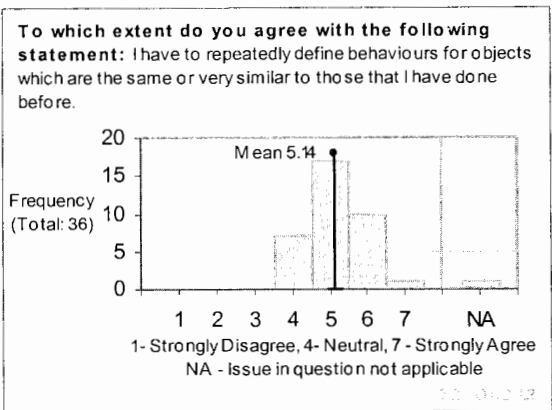


Figure 61: Question 11.2.1.2 from the survey, which queried authors on whether they have to multiply define the same behaviours for similar objects (NA indicates not applicable).

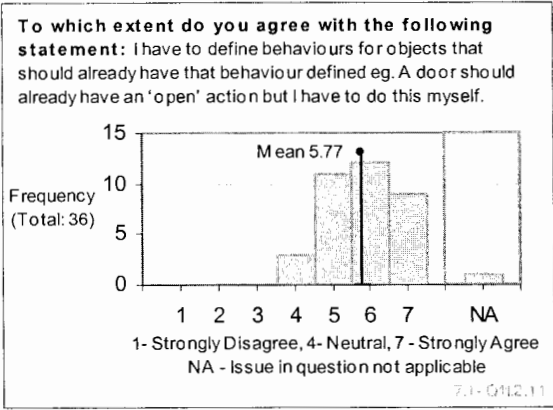


Figure 60: Question 11.2.1.1 from the survey, which queried authors on defining objects' behaviour. (NA indicates not applicable)

P6C3 Difficult or Non-existent Global Environment Specification

When creating a virtual environment, many environmental appearances had to be simulated – appearances such as the time of day or weather. Often there were 'tricks' involved in simulating night time, such as lowering of ambient lights, change of skyboxes and addition of fog. It was preferable for the system to take care of this so that authors can apply the 'night effects' or 'gloomy day' effect to the VE with little effort.

Most of the authors indicated that this issue was applicable to them, with 94% of this group indicated that they had to simulate the appearance of a backdrop or weather condition (see Figure 62). The majority of authors found this process overly complex (see Figure 63).

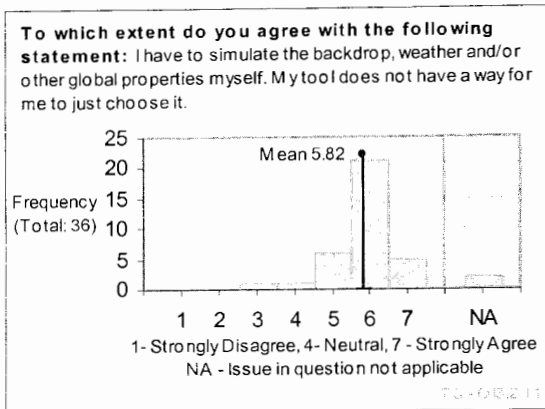


Figure 62: Question 13.2.1.1 from the survey, which queried authors on the how they can add backdrop/skybox and/or weather conditions to their VE. (NA indicates not applicable)

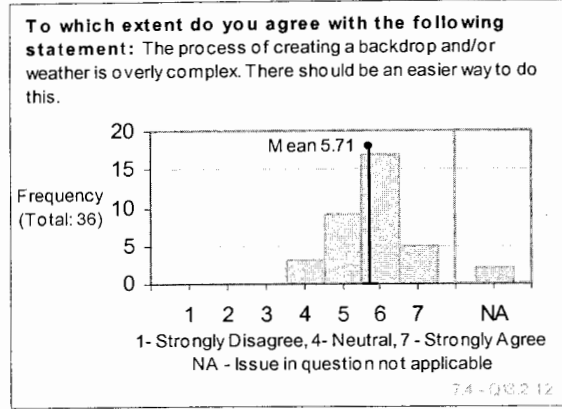


Figure 63: Question 13.2.1.2 from the survey, which queried authors on the complexity of adding backdrop/skybox and/or weather conditions in their VE. (NA indicates not applicable)

P6_{C/AT} Authoring Process

It was clear from the interview that most of the authors felt that the lack of gravity and collision detection lengthened their authoring time. However, most agreed that getting around the two issues was a matter of knowing what to do. Authors with more experience tended to rate this issue as less important, but most saw it as a significant hindrance. It should also be pointed out that this issue affected only certain sections of the interviewed authors and also those who took part in the survey. There were some VE scenarios where both are not as applicable as others. An example of this was the flight simulation-based VE. Overall, surveyed authors mostly ranked this issue as somewhat problematic (see Figure 65a).

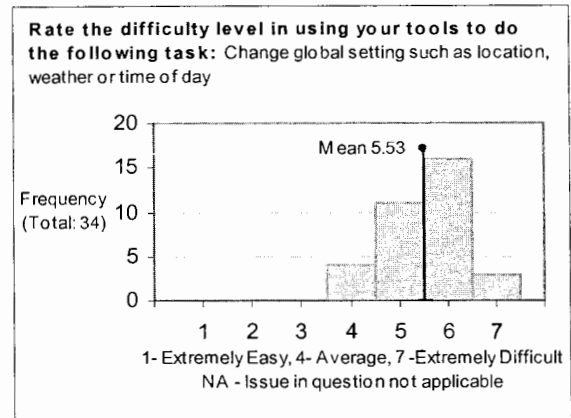


Figure 64: Question 4.2.2.7 from the survey, which asked authors to give difficulty rating for the task of changing global setting such as location, weather condition or time of day.

The second and third issue seemed to frustrate authors more than the first. This was also apparent in the survey (see Figure 65b & c). The issue of object mapping mainly concerned duplication of similar or same interaction specification for many objects and there were many different ways to get around this issue (e.g. Creating a class of objects or a group of objects and define interaction for the class or group). There were also similar workarounds for specifying weather condition and time of day.

However, most authors would see these solutions as advanced or too complex. In the survey more than 88% of authors rated the task of simulating weather or time of day as higher than average difficulty level (see Figure 64).

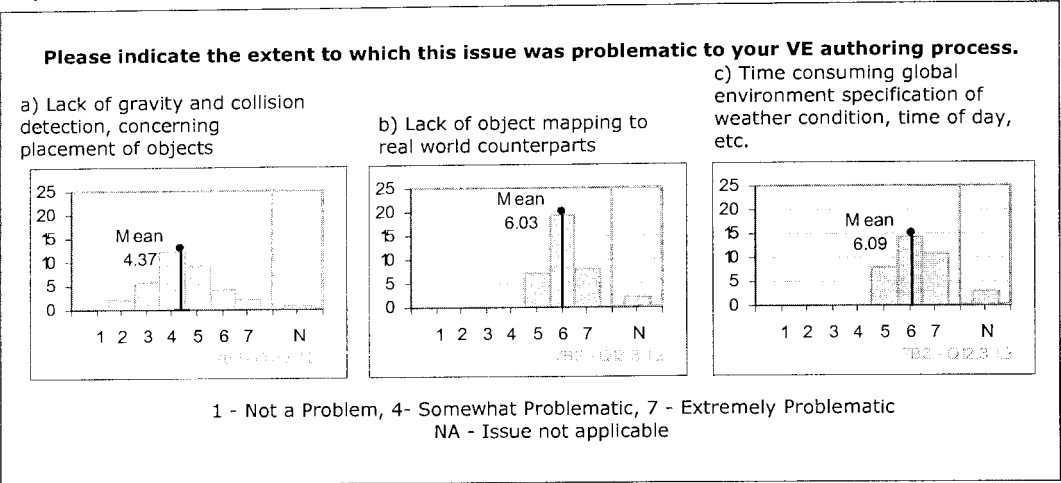


Figure 65: Results from survey Questions 11.3.12, 13.3.1.2 and 12.3.1.2 which enquired the users on the effects on authoring process, by a) lack of gravity and collision detection, b) lack of object mapping to real world counterparts and c) time consuming global environment specification of weather condition and time of day.

P6_{CULT} Learning Process

The common theme running through the three aforementioned issues was that authors' knowledge of the real world was not used to the make the task of authoring easier. Tools that exploited this knowledge should therefore reduce both the authoring time and the learning time.

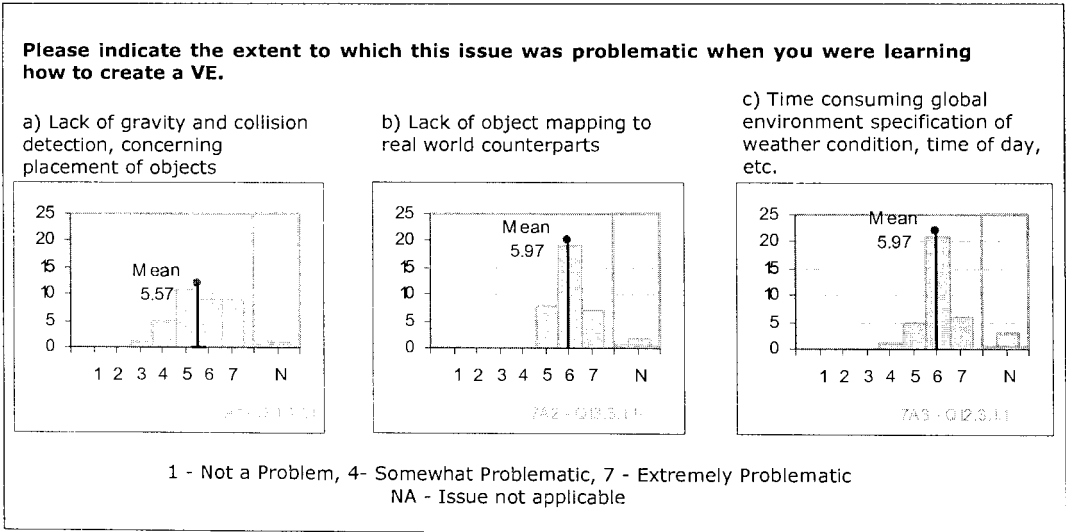


Figure 66: Results from survey Questions 11.3.11, 13.3.1.1 and 12.3.1.1 which enquired the users on the effects on learning process, by a) lack of gravity and collision detection, b) lack of object mapping to real world counterparts and c) time consuming global environment specification of weather condition and time of day.

During the interview, the lack of gravity and collision detection often led to complaints about the three-view point placement interface as it was this interface that most authors relied on to do this task. For many authors, learning to deal with the lack of the two functionalities also meant learning to use the multiple-viewpoint placement interface effectively. In the survey, most authors found this issue to be problematic to their learning process (see Figure 66a), but a lot of them also commented on the difficulty of the placement interface.

In learning to deal with the second and third issue, authors usually learnt how to manipulate their tool to accomplish these tasks. There were some complaints of lack of documentation and how-to documents specifically written on these subjects. The workaround, once learnt, can be applied quite easily, but many authors chose not to go this route as some are regarded as unnecessarily complex or that the way they have been doing things is sufficient.

The survey participants rated this issue concerning object mappings as one of the more problematic ones, with all rating the issue as being more than 'somewhat problematic' to their learning process (see Figure 54b). The third issue of applying weather conditions, time of day etc, received the same average difficulty rating as the second (see Figure 54c).

P6_{C/VE} Overall Quality of VE

For all three areas, authors indicated mid-scale negative effects (see Figure 67). In the interview, there seemed to be two types of authors. The first were those that have learnt how to deal with the issues in their own way without actually overcoming the problems. For example an author would learn to deal with the second issue, by learning how to copy an object and its interaction and modifying the copy slightly.

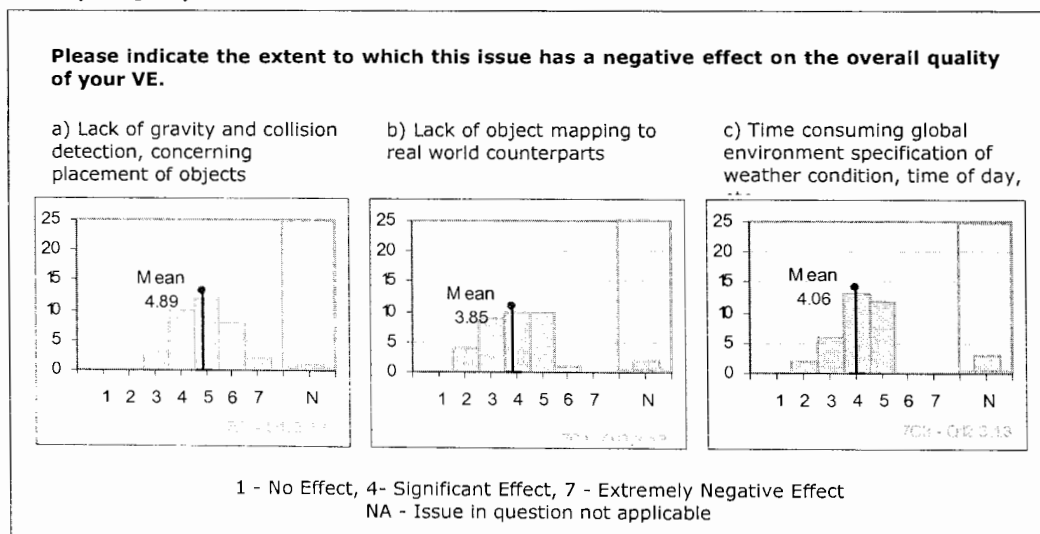


Figure 67: Results from survey Questions 11.3.13, 13.3.1.3 and 12.3.1.3 which enquired the users on the effects on the overall quality of virtual environments by a) lack of gravity and collision detection, b) lack of object mapping to real world counterparts and c) time consuming global environment specification of weather condition and time of day.

The second group would learn the so-called advanced method of dealing with it. In contrast with the author in the previous example, an author in this group would learn how to create a class of object and use that to avoid having to specify similar interactions multiple times. The first group of authors tended to find that the effects of these three issues are minimal, while those from the second group tended towards the opposite.

5.6.2 Analysis

P6_{A1} All purpose VE design tool

The reason most packages do not simulate gravity and collision detection by default could arguably be because the tools were designed for all purpose VEs. Not all VEs need to have these two functionalities turned on during authoring. This reason could also be applicable to the other issues, since providing objects with a specific interaction built-in or providing weather or time setting functionality might limit the tool to only specific scenarios, or at the very least limit the type of the VE that it could build.

P6_{A2} Single-purpose VE design tool

This is almost the opposite of the previous reason. Some VE authoring tools are very specific to a purpose and for that purpose it might not be necessary to have these features. For example, the BUILD tool for creating an architectural fly-through does not provide gravity or collision detection by default, because its VE, the fly-through, does not have to test if the end-user were flying through a wall or floors.

P6_{A3} object libraries are time consuming and commercial unviable to develop

Different users expect any one object to perform different actions. For example, some authors would like a door to open in one direction, others wanted them to slide; some thought that doors should be lockable. It is unrealistic to try to please everyone, but research on what the reasonable interactions/actions are could be time-consuming. Add to this the desire of tool-makers to envelop as many types of objects and thus as many types of authors as possible, creating a viable object library could be very expensive.

5.7 Interaction Specification Interface (P7)

So far, the problems that have been discussed consist mostly of spatial aspects of environment creation. Interaction arguably makes or breaks virtual environments, but yet there seems to be no other effective way to specify⁴ complex interaction other than via scripting or programming language-like methods. There were two main types of interaction specification used by authors who took part in the interviews:-

1. **Scripting/Programming Language:** Tools that claimed to cater for all kinds of possible scenarios of VEs usually rely on a scripting/programming language for interaction specification. With this type of tool, authors specified how and where an action should be performed by *programming* this information in. Some tools provided the ability to do this in the tool, usually via a window, while others required that this be done externally. There were also variations in the compilation process, with some requiring none, while others required a separate and lengthy compilation process.
2. **Simple Event-based Specification:** This type of software package allows authors to select a sequence of actions that should happen for a particular event. For example, in the BUILD tool, an author could specify that a door be swung open when a camera gets close to it or at a certain time period. The actions that can be done, and the conditions that should be tested, are often very limited. For more complex interactions, authors would have to program new actions and/or conditions.

5.7.1 Comments

P7_{C1} Tools require extensive Programming Knowledge

In the interview, two of the authors (the architects) indicated that their tools allow for simple event-based specification, but they often had to resort to programming for more complex interactions. In the survey, most authors, 86%, indicated that they specifically have to learn a programming/scripting language (see Figure 68).

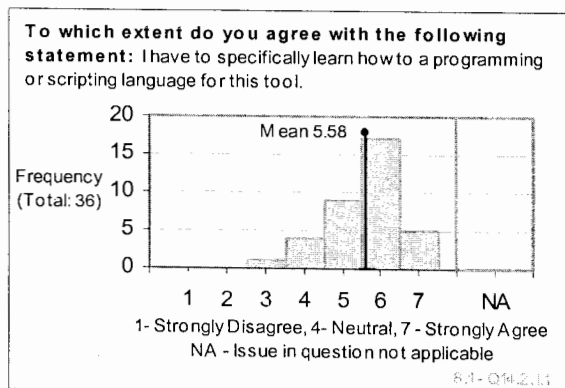


Figure 68: Question 14.2.1.1 from the survey, which queried authors on whether they had to specifically learn how to program in order to create a VE.

⁴ Interaction specification refers to the process of defining relationships between different objects in the environment and between objects and the environment.

P7C2 Avoidance of complex interactions

All but one of the authors interviewed indicated that adding behaviours or interactions to objects by programming or scripting was the one of the more difficult tasks in the authoring process. This was confirmed by the authors in the survey with 72% agreeing that they found the task difficult (see Figure 69).

Additionally, all of the interviewed authors agreed that they sometimes avoid scripting or programming more complex actions. Most of the reasons given were that they feel that it would either be too difficult or time consuming. In the survey, 88% of surveyed authors indicated that they too avoid adding complex action, feeling that programming or scripting them might be difficult (see Figure 70).

P7C3 Re-enforce multiple tools/modes mentality

The majority of the interviewed authors and 86% of the surveyed authors (see Figure 71) used tools that have an interaction editor (for entering scripts or code) appearing as a separate application (5/8 of those interviewed), or in a separate window/pane (2/8 of those interviewed). This separation could further reenforce the multiple tools/modes mentality and its associated effects.

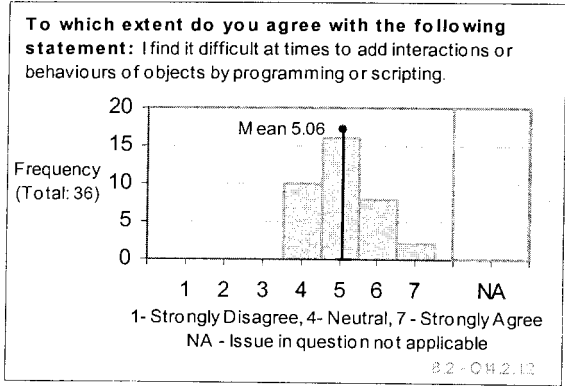


Figure 69: Question 14.2.1.2 from the survey, which queried authors on the difficulty of adding interactions or behaviours by programming or scripting.

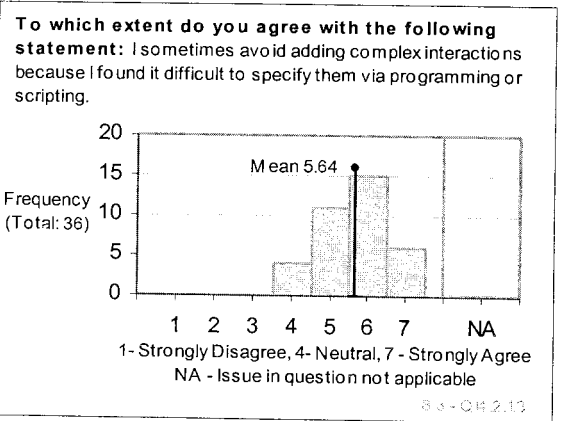


Figure 70: Question 14.2.1.3 from the survey, which queried authors on whether they avoid adding complex interactions because of difficulty with the specification.

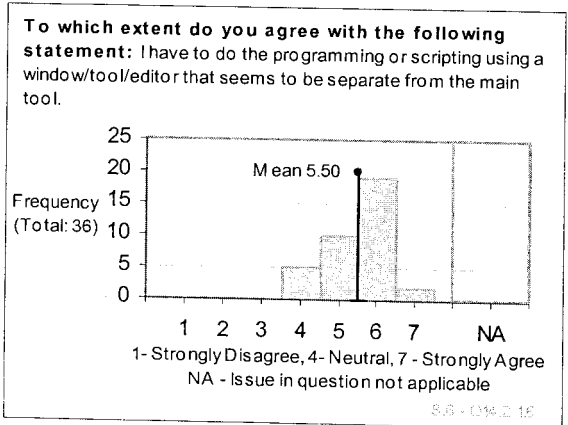


Figure 71: Question 14.2.1.6 from the survey, which queried authors on whether they do the programming/scripting in a separate interface.

P7C4 Re-enforce two steps mentality

Many of the authoring tools that were used by interviewed authors employed compiled programming languages such as C++, as compared to some which used interpreter-based scripting languages. Use of languages that required compilation led to delays since any interaction code had to be compiled and then run before they could be viewed inside a VE. This delay could reenforce the multiple steps mentality outlined earlier in 5.2 Compile and View Model. Most of the authors who took part in the survey also agreed that the compilation step made the authoring process seem tedious and time-consuming (see Figure 72).

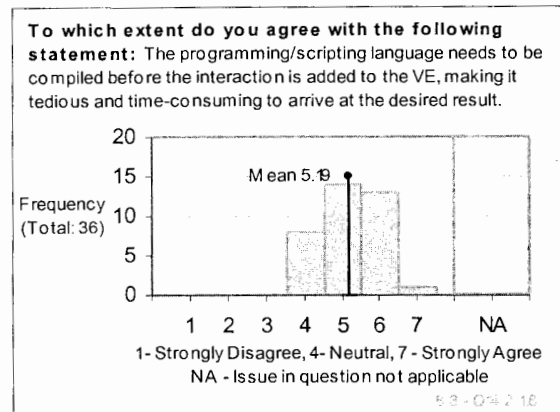


Figure 72: Question 11.2.1.2 from the survey, which queried authors on problems stemming from the compilation aspects of their interaction specification.

P7C5 Object Selection Problem

In section 5.5 Selecting and Editing Objects, the problem of indirect selection of objects were presented. The scripting and programming interface often meant that authors had to deal with remembering objects' names and typing in the correct one. In the interviews, some authors' complained that the separate editor window or application made it difficult for them to connect different objects that they were trying to linked in an interaction. Often they had to switch back between the two interfaces making it tedious and time consuming. Survey participants also agreed on this issue (see Figure 73).

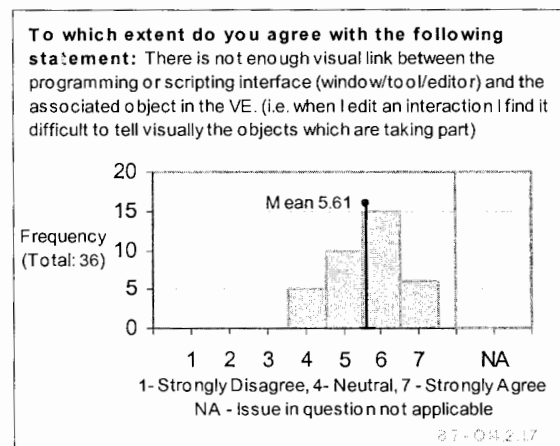


Figure 73: Question 14.2.1.7 from the survey, which queried authors about the visual linkage between objects which are in an interaction.

P7C6 Locating and visualising interactively-linked objects

For most tools that separated interaction specification interface from the VEs, specifying interactional relationship amongst objects were usually entirely contained within the interaction interface. In the worst case, this relationship was only reflecting programmatically in the code or script. In the interviews, authors using interaction specification interfaces that were separated from the rest of the authoring package all had no other visual access to interactional relationship, except

through code. The problem with having to go through code to find all the interactional relationship was primarily that things became very confusing with increasing numbers of interactions. It was often unclear what each portion of the code was doing. One author stated that often, if the scenario's interactions were not well thought out at first, it could get very difficult towards the end to finish off.

Visualising interactional links could become quite important in developing large scenarios. This would give the authors the ability to see the overall picture of the VE and the relationship between different objects. Most interviewed authors found it frustrating if the interactions were not sufficiently specified beforehand. This is no surprise considering the difficulty involved. In the survey, 86% of all participants rated the task of finding objects that take part in an interaction as higher than average difficulty (see Figure 74). This trend, and many comments made in the survey, reflected both the difficulty in reading the relationship from the code and the frustration involved in mapping between objects' representation in interaction code and the VE itself.

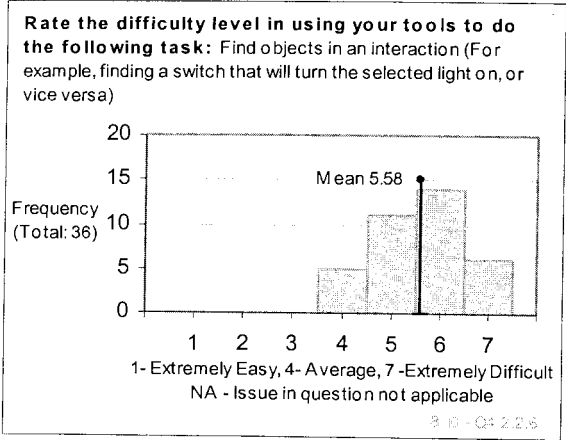


Figure 74: Question 4.2.2.6 from the survey, which asked authors to rate how difficult it is to find objects that are in an interaction.

P7_{CAT} Authoring Process

In addition to the trend that most authors found adding interactions by programming, or scripting, difficult (see Figure 75), most authors also found that the process itself takes a long time to do (see Figure 76). Compound this with the problem outlined earlier in 5.6 Lack of Real World Mappings, with authors reporting that they would rather duplicate codes than explore other means, and it was no surprise, therefore, that many authors reported that they spent a lot of time programming or scripting similar or slightly varied interactions (see Figure 77).

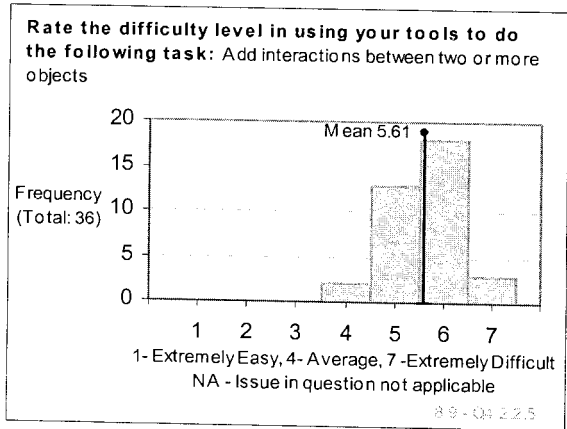


Figure 75: Question 4.2.2.5 from the survey, which asked authors to give difficulty rating for the task of adding interactions between two or more objects.

All these factors contributed negatively towards the authoring process; with as much as 88% of authors in the survey ranking the programming or scripting-based interaction specification as more than somewhat problematic to their authoring process (see Figure 78 middle).

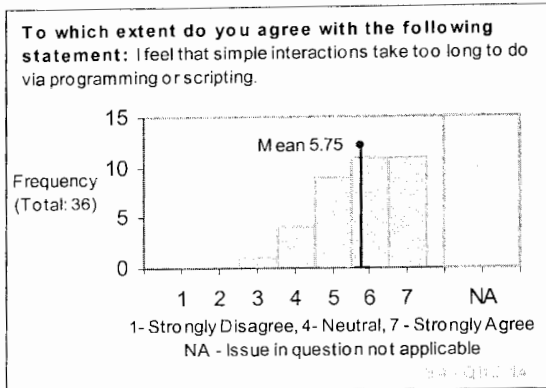


Figure 76: Question 14.2.1.4 from the survey, which queried authors on time taken to create a simple interaction via programming. (NA indicates authors not using this type of interface)

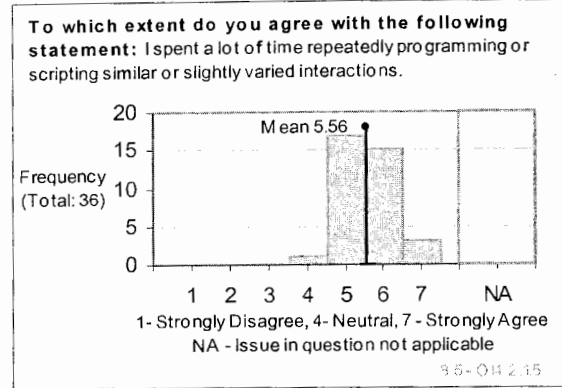


Figure 77: Question 14.2.1.5 from the survey, which queried authors on whether they spend a lot of time repeatedly programming similar interactions (NA indicates authors not using this type of interface)

P7_{CLT} Learning Process

During the interviews, most authors indicated that the steepness of their learning curve was dependent on how much they knew about programming beforehand. Those that have prior knowledge had less problem learning to script or program interactions. However, most of those interviewed had started with minimal programming knowledge and had struggled to deal with this aspect of authoring. Most made quite clear their frustration with such a system and would prefer a simpler way, even if it meant that they are limited to simpler interactions. In the survey this was also the case with most authors tending towards the extreme negative (see Figure 78 left).

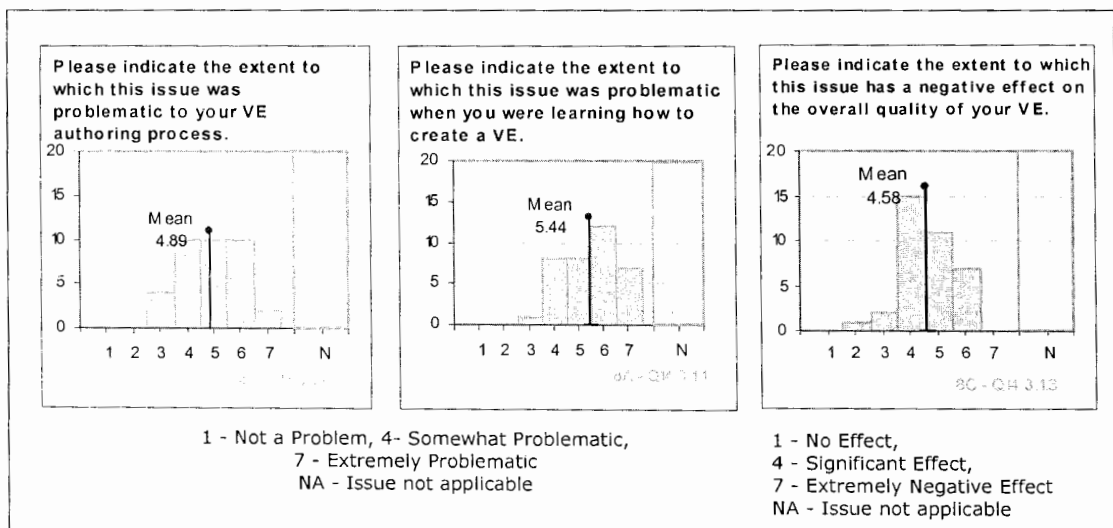


Figure 78: Results from survey Questions 14.3.1.1-14.3.1.3 which queried the users on the effects on learning, authoring and quality of VE, using programming or scripting to create interactions.

P7_{C/VE} Overall Quality of VE

Most authors indicated that they often avoided complex interactions because of the perceived difficulty involved with programming or scripting. Some of this action might result in a lower quality of VE, especially in the mind of these authors. Some were content to achieve what they could without programming or scripting, while others struggled to even get the simplest interactions correct. Some of the authors interviewed rated scripting and programming interactions as the most influential factor to their VE quality. 91% of the surveyed authors agreed that the issue also has at least significant effect on the overall quality of their VEs (see Figure 78 right).

5.7.2 Analysis**P7_{A1} Programming provides most functionality**

In almost all packages that were used to create VEs, the process of specifying the behaviour of objects in the environment was done by means of programming. While a programming language may allow authors to create any type of interaction, it makes the process very difficult for users with little or no prior programming knowledge. There have been many alternative approaches, of which the simple event-based specification is one. While such an approach can provide a much simpler way to specify interactions, it is limited in the types of interaction that can be specified.

P7_{A2} Parallels between Software and VE Development

VE authoring tools have not been sufficiently popular to ensure their spread to the more non-programming user base. In many cases, the end-user of the tools (in this case the authors) were not kept in mind when designing. In many ways designing interaction in a VE is like programming and thus programming/scripting languages seem like the best choice.

5.8 Time Manipulation (P8)

Creating a good virtual environment involves managing objects' spatial and temporal interactions with other objects and with the environment itself. Most tools give a lot of focus on doing the spatial portion very well, while providing minimal support for the temporal portion. There are many time-related problems with the authoring tools in use by authors mentioned in both the interviews and the survey. Some of these are outlined below:

5.8.1 Comments

P8_{C1} Time wasting due to Lengthy Content

Often in a VE, there is a long sequence of action such as a long dialog sequence. As authors frequently previewed their VE, this often resulted in a long wait. In one of the cases, the author became very frustrated as there was a very long sequence right at the beginning and he had to sit through it every time he previewed the VE. What was desirable here would be the ability to fast-forward through time so that unimportant events could be skipped. Many authors also suggested that a rewind facility be made available as well, so that authors could go back to an earlier time to try a different alternative in the storyline, or perhaps to re-observe different parts of the same environment. Time manipulation, should it be introduced, should only be available to authors during authoring or preview time.

P8_{C2} Difficulty in Previewing Simultaneous Actions

Another common situation was when two or more simultaneous actions had occurred (as part of an interaction) and the author had to verify that this had happened. The problem was that she may not be able to (or may not know how to) be in two places at once. While the rewind facility might also be useful

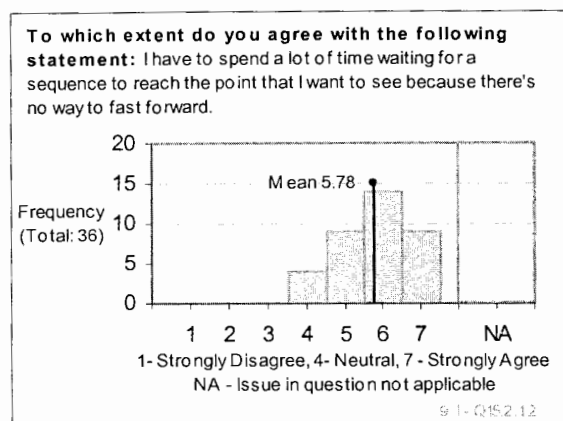


Figure 79: Question 15.2.1.2 from the survey, which queried authors on the time spend waiting for a sequence to take place because there is no fast forward functionality.

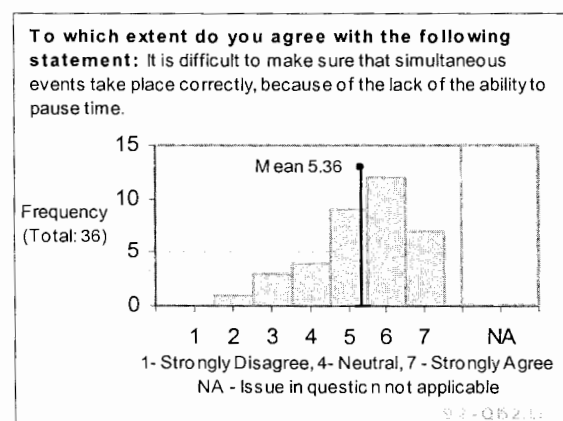


Figure 80: Question 15.2.1.1 from the survey, which queried authors on the difficulty in checking that simultaneous actions occur correctly.

in solving this issue, the ability to pause the environment could also be advantageous. In addition to being able to be in two places at the same time frame, authors would essentially be able to examine a snapshot of the environment.

P8C3 Difficulty in Creating Time-based Sequences

Most of the authors interviewed indicated that they had to, at one time or another, create a time-based sequence of events. An example involved a sequence of dialog, where an animated character would *say* one piece of dialog for a certain time period, after which another would be played. A few tools did provide a timeline-based interface similar to that found in a lot of media editing programs such as Adobe Premier or Macromedia Director. While these interfaces were found to be effective and relatively easy to learn, some authors commented on the complexity of this type of interface when there were simultaneous events on the timeline. The difficulty rating for creating a time-based sequence, as given by authors in the survey, is shown in Figure 81. Most (5/8) of the authors who indicated neutral and a few who scored 5 (3/11), commented that they had a timeline-based interface.

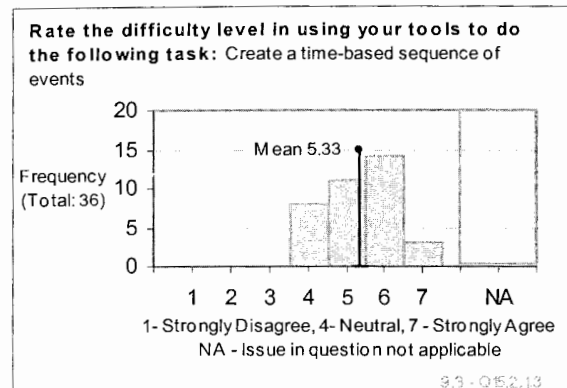


Figure 81: Question 4.2.2.8 from the survey, which asked authors to rate how difficult it is to create a time-based sequence of events.

P8C4 Other Reasons for Pause, Rewind, and Fast-forward Functionality

In addition to the reasons given in both P8C1 and P8C2, authors also indicated that they would find the ability to pause, fast forward, or rewind while authoring useful. The result in the relevant question in the survey is shown in Figure 82. There were a few authoring tools that let authors pause the simulation of the VE during previewing (but not at actual run-time).

A few tools with time-manipulation facility during preview time included the following features:

1. Pausing the environment to select, manipulate and/or view the state of a moving object.

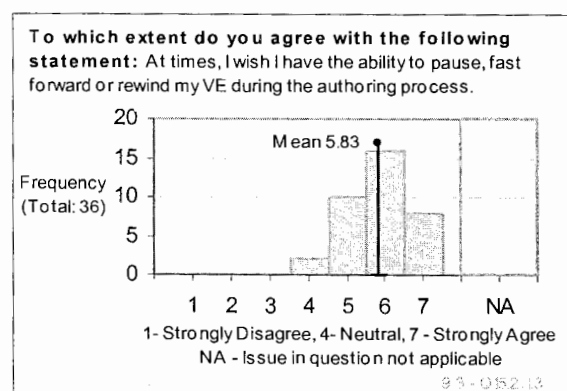


Figure 82: Question 15.2.1.3 from the survey, which queried the authors on whether they would like the pause, fast forward and rewind functionalities.

2. The ability to go to a certain point in the environment, with pre-programmed user-based interactions so authors did not have to sit through everything after many restarts.

P8_{C/AT} Authoring Process

Time wastage seemed to be the major consequence of not being able to manipulate time during the authoring. Most complaints were related to the restarting process when after starting authors had to go through all the sequences again. Sample comments included *'I have to sit through everything every restart.. it drives me insane..'* and *'I could make a cup of coffee every time.'* Other complaints were related to difficulties involving creating time-based sequence which were usually also time-wasting. Figure 84 (top) shows the relevant effect rating.

P8_{C/LT} Learning Process

With time manipulation issues, most authors interviewed indicated that they had to learn how to get around the problem. Most of the time patience was what was needed. Comments from the survey also suggested frustration from the delay. Some authors developed techniques to deal with this over time, including breaking down their VE and action sequence into smaller, more manageable chunks, but these techniques were often dependent on the tool and/or the nature of the scenario being developed.

P8_{C/VE} Overall Quality of VE

A VE often need to be tested through the entire possible scenario before it could be declared ready to use by the end-user. Often this was done by the authors themselves running the environment to try out several options. The best scenario was for the authors to try all the possibilities and depending on the complexity of the VE, this may or may not be possible. The testing process would be more difficult if authors had to restart from the beginning every time an interaction branch needed to be retraced. Often due to time-restraints, not all possible branches were tested. It is this point that might result in the virtual environment not working as expected. Some

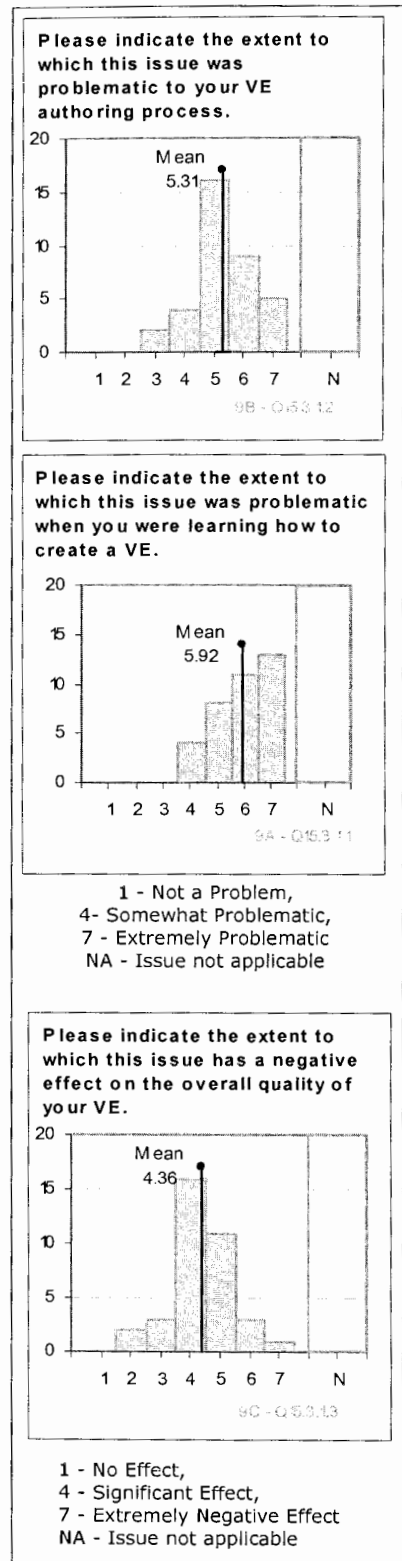


Figure 84: Results from survey Questions 15.3.1.1-15.3.1.3 which queried the users on the effects on learning, authoring and quality of VE, by the lack of time manipulation functionality.

of the authors in the interview have indicated that this could potentially lower the quality of a virtual environment. Most others, though, suggested that frustration with having to restart and wait around could cause the virtual environment to not result as expected. A few comments in the survey along this line included *'If I haven't had to wait around, I could probably do better.'*

5.8.2 Analysis

P8_{A1} Lack of time-manipulation functionality

The lack of a time-manipulation facility can be attributed to the fact that it is undesirable for the consumer to manipulate time during viewing and since most authors are forced to only view their creation through the same viewing interface as the end-user, they also have no time-manipulation facilities.

P8_{A2} No underlying support

In order to support time manipulation, every single time-based action, including animation and audio playing, needs to be based on a single clock. This is not the case in some authoring tools, with some handling animation using a different clock to the one handling the interaction. In some cases, the speed of animation is not time-dependent, but rather frame-rate dependent. These factors contributed to making time-manipulation facilities difficult to implement in existing tools.

5.9 Summary

The top eight issues with virtual environment authoring tools used by a relatively experienced group of content-expert authors were presented. We also presented comments made by those who took part in the interview and the survey on each issue, as well as an analysis of the possible reasons why each is present in a number of virtual authoring tools. Table 1 contains the summary of descriptions of issues, comments, ratings and analysis.

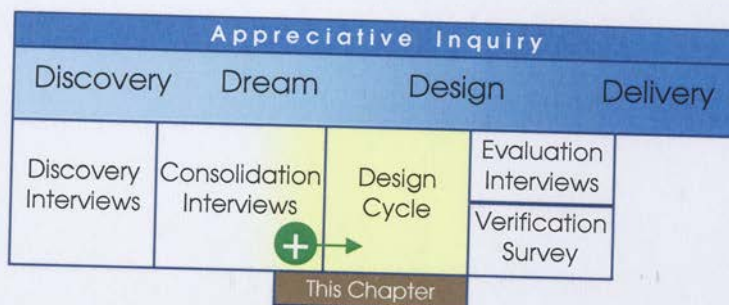
Table 1: Summary of the top eight issues related to using virtual environment authoring tools, comments on each by a wide group of authors, and analysis of possible reasons why each is present in a number of virtual authoring tools used by these authors.

Issue and Description			Score and Rankings					
			Authoring		Learning		VE Quality	
			Mean	Rank	Mean	Rank	Mean	Rank
P1 Multiple Tools A number of tools may be used in VE authoring process, usually one for building environment and one for specifying interactions.	Comments	P1 _{C1} Confusing about different tools P1 _{C2} Incompatibility issues amongst tools P1 _{C3} Multiple steps mentality P1 _{C4} Unnecessary saves or backups	5.80	4	4.80	9	4.27	7
	Analysis	P1 _{A1} Combination of existing tool P1 _{A2} Separation of tasks						
P2 Compile & View Model: VE needs to be compiled or built before the end-result can be seen. This process can be quite lengthy.	Comments	P2 _{C1} Different representations P2 _{C2} Lack of immediate Feedback P2 _{C3} Build All then View mentality P2 _{C4} Different controls for edit and view modes	5.37	6	5.30	8	5.53	1
	Analysis	P2 _{A1} Difference in platforms P2 _{A2} Difference in interfaces P2 _{A3} No Real-time update P2 _{A4} Time-frame issues						
P3 Mixed 2D/3D Interface: Tools contain mixture of 2D/3D interfaces.	Comments	P3 _{C1} Pointer confusion P3 _{C2} Constant switching between mindset P3 _{C3} Cluttered interface P3 _{C4} Reduced viewing area P3 _{C5} Delayed update P3 _{C6} Build All then View mentality P3 _{C7} Difficulties in directly editing an object's attributes P3 _{A1} Extension of existing interfaces	4.31	10	4.14	10	2.78	10
	Analysis							
P4 Multiple View-points Object Placement Interface: Placing objects involve authors monitoring different viewpoints	Comments	P4 _{C1} Confusing interface P4 _{C2} Cluttering interface P4 _{C3} Difficulties in moving, placing and locating objects	6.22	1	5.78	5	4.44	5
	Analysis	P4 _{A1} Mixture of interfaces P4 _{A2} Lack of gravity and collision detection						

Issue and Description			Score and Rankings					
			Authoring		Learning		VE Quality	
			Mean	Rank	Mean	Rank	Mean	Rank
P5 Selecting and Editing Objects: Problems with indirect object selection.	Comments	P5 _{C1} Confusing interface P5 _{C2} Difficult to use P5 _{C3} Keeping track of selected object saves or backups	5.69	5	5.93	3	4.86	3
	Analysis	P5 _{A1} Adaptation of existing interfaces						
P6 Lack of Real World Mappings: Tools not reflecting real environment	Comments	P6 _{C1} Lack of gravity & collision detection →	4.37	9	5.57	8	4.89	2
		P6 _{C2} Minimal object mapping →	6.03	3	5.97	1=	3.85	9
		P6 _{C3} Difficult or non-existent global environment specification →	6.09	2	5.97	1=	4.06	8
	Analysis	P6 _{A1} All purpose VE design tool P6 _{A2} Single-purpose VE design tool P6 _{A3} Time consuming and commercial unviable to develop object library						
P7 Interaction Specification Interface: Problems with specifying interactions	Comments	P7 _{C1} Tools require extensive programming skill						
		P7 _{C2} Avoidance of complex interactions						
		P7 _{C3} Re-enforce multiple tools/modes mentality						
		P7 _{C4} Re-enforce two steps mentality	4.89	8	5.44	7	4.50	4
P8 Time Manipulation: Lack of functionality to manipulate time during preview of VE.	Comments	P7 _{C5} Object Selection problem						
		P7 _{C6} Locating and visualising interactively-linked objects						
		P7 _{A1} Programming provide most functionality						
		P7 _{A2} Parallel between software and VE development						
	Analysis	P8 _{C1} Time wasting due to lengthy content						
		P8 _{C2} Difficulty in previewing simultaneous actions						
		P8 _{C3} Difficult to create time-based sequence						
		P8 _{C4} Other reasons for pause, rewind, and fast-forward functionality	5.31	7	5.92	4	4.36	6
	Analysis	P8 _{A1} Lack of preview mode						
		P8 _{A2} No underlying support						

6

Solutions and Resolutions



In this chapter, we present the solutions that were derived from three sources: positive data from the discovery and consolidation interviews, author's input and research. They form the result of the dream and design phases of the appreciative inquiry methodology.

Specifically, the solutions were designed to resolve the interface issues outlined in the previous chapters. There are cases where it was not possible to completely resolve the issues, as some were in conflict and solving one exacerbated another.

We begin with the descriptions of each issue and how they contribute to the resolution (or at least minimisation) of the different issues. We end off by going through each issue and discussing how the different solutions address each.

6.1 Solution Overview

The system that was arrived at after the consultative process is presented here. Below, each sub-module is briefly outlined. In the later sections they are explained further, with a description of which issues (or part thereof) were resolved.

- VE clock:** We employ a clock to keep global time for the entire environment. There is an initial state with captures every object's position and internal states at time index 0. There is no concept of run or build, the closest equivalent to this is the 'restart clock' functionality where every object is reset to its initial state at time index 0. Figure 85 contains screenshots of the temporal manipulation control of our tool. See 6.2.1 Temporal Manipulation for more detail
- Unified WYSIWYG Interface:** The virtual environment acts and reacts in the same way whether the environment is edited by authors (authoring profile) or run by the consumer (consumer profile); the only difference will be that the authoring profile will have certain control functionality available, such as time and object placement and behavioural manipulations. The same interface is used to handle both environmental building and interaction specification. Furthermore, during authoring, authors can *fly* and have a bird's eye view of the environment (Figure 86). See 6.2.2 Unified WYSIWYG Interface for more detail.
- In-environment interface:** To over-come the 2D/3D separation, and also to be compatible with most hardware platforms, all user interfaces should appear within the environment itself i.e., in 3D. Direct object selection and manipulation is also possible with this interface. Figure 87 contains various screenshots of different parts of the interface; see 6.4 In-Environment Interface for more detail.



Figure 85: Screenshots of the temporal manipulation control, from first person view (top) and from bird's eye view (bottom)

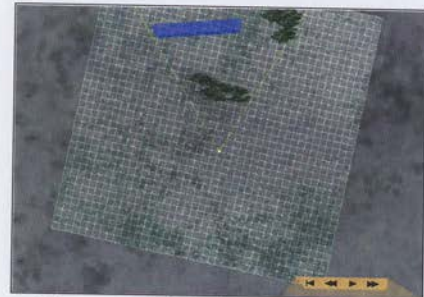


Figure 86: View from the top

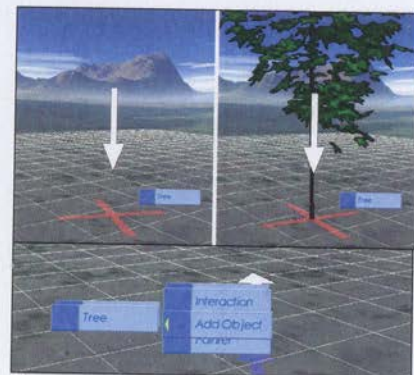


Figure 87: Various aspects of the in-environment interface

- **Interaction System:** Different types of actions and interactions occur in the confines of a virtual environment and its authoring tool. The interaction system handles both authoring actions, VE-based actions, as well as those generated by independent monitors⁵. The system also handles record keepings for undoing and rewinding. See 6.3 Interaction Subsystem for more detail.
- **Object-based interaction specification Interface:** An interface for direct object selection which can be used to manipulate objects as well as specify interactions. Figure 88 shows some screenshots of the different aspects of the interaction specification interface; see 6.5 Object-based Interaction Specification Interface for more detail.
- **The World and Avatar Objects:** The world object represents the global environment and is used to keep track of data such as time, and weather condition, amongst others, while the avatar object is used to represent the end-user. See 6.6.4 The Avatar Object (see Figure 89) and 6.6.5 The World Object for more detail.
- **Advanced Object Library:** A library containing objects and behaviours which can be used by authors to populate the environment. Each object reflects its counterpart in the world and has behaviours built-in which can then be used to specify interactions. For more detail see 6.6.2 Object Library.

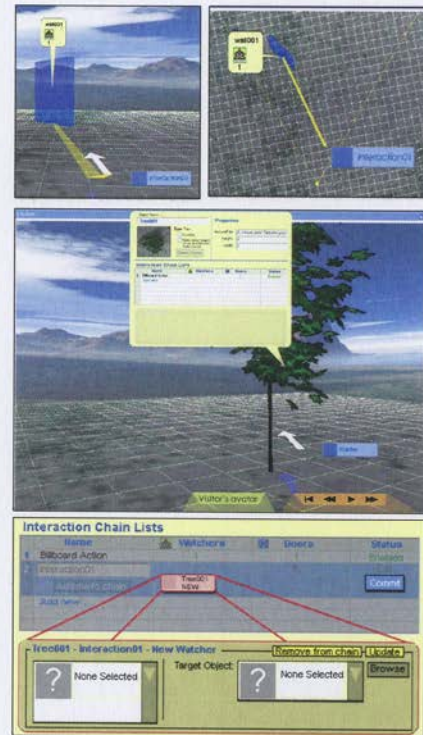


Figure 88: Interaction specification interface: Creating interaction from first person view (top-left), from bird's eye view (top-right); editing an object's attributes (middle) and detailed interface for specifying interaction (bottom)



Figure 89: Representation of avatars from different viewpoints

⁵ Independent monitors continuously watch for conditions and from there generate further actions. An example is the gravity simulator, which constantly watches for applicable objects, and moves them towards the ground.

6.2 VE clock & Unified WYSIWYG Interface

6.2.1 Temporal Manipulation

Simple VCR interfaces are often used in software where no media authoring is involved. For example Windows Media Player, Real Player and Winamp use this interface to allow users to move backward and forward during playback (see Figure 90).



Figure 90: VCR-like control interface for (from top) Media Player, WinAmp and Real Player.

Many temporal manipulation interfaces involve timelines. Simple examples are those seen in video editing and animation software. They have interfaces where users can move a time index, often represented by a slider along a trackbar. Examples include Microsoft Movie Maker and VirtualDub video editing package. The interface is used extensively in creating media, with users able to drop videos, audios and other media directly onto the timeline. Other actions can be done on the timeline such as dragging the representation of the media to fit into a timeframe.

However, timeline-based interfaces are suitable for static, non-branching videos or animation. For branching media, a single timeline interface can be used to present activities along one selected event timeline i.e. events that would happen due to the current state. Moving the static timeline will take users to the future state of that particular timeline. Examples of such system include Macromedia Director 6. It has been suggested that the use of timeline interface in branching media is confusing [88], often resulting in what Wolber calls 'spaghetti timeline'. This is a play on the phrase 'spaghetti code', describing the case where authors pile on more interactions along the same timeline in a disorganized fashion – often with disastrous results.

Considering the need for temporal manipulation in VE authoring

The main temporal manipulation requirement for our authors concerns the previewing of the VE. All three main comments P8_{C1}, P8_{C2} and P8_{C3} are concerned with the ability to make sure that the VE acts in the way the author has conceived it. It is for this reason that a VCR-like interface (see Figure 91) was chosen over a timeline-based interface. However, it was discovered that authors would like to be able to perform authoring tasks (such as adding objects or creating interaction) while manipulating time. However, this introduces complexity in the question of whether the authoring action is linked to the current time index e.g. when authors add an object at a certain time index, does it mean that the object only

appears at that time during normal running of the VE? The question is looked at in more detail later in 6.3 Interaction Subsystem.

Recording VE

The issue of recording a VE in order to be replayed has been implemented in many systems. In the CAVERNsoft system [37], a persistent VE is achieved partly via the recording of the avatar's movements and audio. As part of the COVEN project, DIVE was extended to include a logging facility to record activities within a VE [24]. The temporal link-based system proposed by Greenhalgh *et al.* [26], was created with ability to replay past or recent recordings of a VE within another VE. We used a similar journaling system to keep track of activities that have happened in order to provide the rewind facility to our authors. (This journaling system is looked at further in 6.3 Interaction Subsystem.)

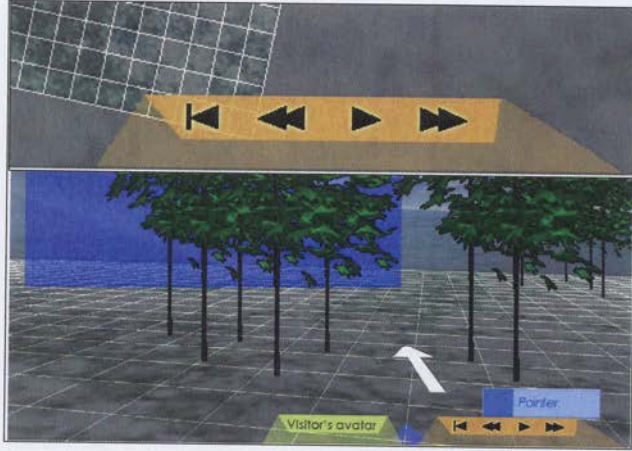


Figure 91: Screen capture from VEBUILDER II, showing the WYSIWYG interface and the time manipulation console.

VE Clock

To fast forward events in the VE, we use a VE clock which keeps time for the entire virtual environment. A global clock has been used in many multimedia systems [2], such as in video editing software to ensure synchronisation between the different types of media (usually videos and audio) whenever the time-index is shifted. The VE clock has the following features:

- The clock starts at zero and remains at zero until activated by the author (or end-user).
- By default, it runs at the same rate as a real clock. Authors can speed up or slow down the clock (This is an example of an authoring action, see later).
- All possible non-instantaneous actions are run with respect to the clock. Such actions include model animation and audio playback. All non-authoring action can only take place when the clock is running.

- A snapshot of the environment can be saved, storing all the objects' states. These states can be restored at a later stage. At time index zero, one such snapshot is saved and can be restored at any time.

6.2.2 Unified WYSIWYG Interface

By allowing authors to make changes dynamically within the environment, authors can evaluate the changes straight away. This can be achieved by introducing a WYSIWYG interface, where the virtual environment is the same during both authoring and viewing (at a later stage by the end-user). There have been many VE authoring systems taking this approach, including [89] and more recently the *explorative* VE construction method by Seo *et al.* [57]. This system uses an interactive kernel that allows authors to configure the aspects of form, function and behaviour separately and dynamically.

Dynamic behaviour specification using a scripting language has been used in many tools. This allows authors to rapidly create and change interactions without restarting the environment. Alice [71] is the forerunner in this aspect, with many gaming engines also adopting this method.

Taking a cue from these systems, ours was based on a WYSIWYG interface in the sense that as authors build their VE, what they see is the same as what an end-user visiting their VE would see⁶. In order to facilitate this, we have chosen to *add* authoring interfaces onto the 3D VE i.e. the interface will exist *inside* the environment, not unlike those suggested by Yang *et al.* [89]. We called this kind of user interface an *in-environment interface*. This was used in the selection interface, with the cursor as well as the menu system. More detail can be found in the section 6.4 In-Environment Interface. We also adapt the WYSIWYG aspect to our interaction specification system, whereby the author is able to create interactions and see their effect immediately after the creation process in a similar way to a scripting-based system such as Alice [71]. Our interaction interface will be discussed later in 6.5 Object-based Interaction Specification Interface, where our choice of the name will become clearer.

In our system, authoring the VE is only possible with an author profile. Non-authors can enter the VE and will be bound by all the interaction specifications in effect within the VE. They will also be under the influence of the VE clock with no ability to manipulate time. Authors, on the other hand will be able to access various authoring interfaces, including adding objects, specify interactions and manipulating time. However, concerns were expressed about managing authoring rights and how this could be controlled.

⁶ In fact, the same interaction and display engine are used for both the authors and the end-user.

Interchangeable Viewpoints

Two interchangeable viewpoints are accessible to the author. One is the bird's viewpoint, allowing authors to fly around in the environment and the other is from the point of view of the end-user i.e. first person viewpoint. The transition between the two view points is smooth in order to preserve context [60] and minimise the possibility that authors will get lost during the change.

6.2.3 What do VE Clock & Unified WYSIWYG Interface solve?

- **P2⁷ Compile & View Model:** The environment is *always running* so there is no need for a compilation process. Any changes are reflected straight away whether or not the clock is running. This would affect all issues commented on by the authors:
 - **P2_{C1} Difference in representation:** Since there is only one representation, this is no longer an issue.
 - **P2_{C2} Lack of immediate feedback:** Changes are reflected immediately but there may not be any reaction if the clock is not running.
 - **P2_{C3} Multiple steps mentality:** By removing the need to compile, authors are not forced to go into the compile/build loop, leading to elimination of this issue.
 - **P2_{C4} Different Controls in edit and view modes:** Similarly for this issue, the two different modes are combined into one interface, with the authoring one available on top of the end-user-only one.
- **P4 Multi-pane placement interface:** In the interviews, authors indicated that having only bird's eye and first person viewpoints should be enough for them to accurately place objects. By presenting these two viewpoints, we should be able to overcome the problems associated with the multi-pane placement interface and yet allow for reasonable accuracy. Furthermore, the overall view of the environments will prove useful in visualising objects that take part in an interaction⁸.
- **P8 Time Manipulation:** By ensuring that every non-instantaneous action is based on the same VE clock, we can give authors the ability to fast-forward by simply changing the speed of the clock (> 1:1s faster, < 1:1s slower or 0 for pause). The fast-forward and rewind features work in combination with the interaction subsystem to track predictable events and actions.
 - **P8_{C1} Time wasting:** Lengthy content can be fast-forwarded through.

⁷ In this chapter we will be referring to authors' comments using the same notation that was used in the previous chapter. Please refer to the previous chapter for more detail if the comments presented in this chapter are not sufficient.

⁸ See 6.5 Object-based Interaction Specification Interface for more detail.

- **P8_{C2} Previewing simultaneous action:** The environment can be paused or slowed down, give the author a chance to look around to verify that any specific action has occurred.
- **P8_{C3} Pause, rewind, fast-forward functionalities:** These are available through the interface.

Below is a list of other issues that are also resolved partially in combination with other functionality to be introduced later.

- **P1 Multiple Tools:** This is achieved by providing a unified interface for building environments and interaction specification.
- **P3 2D/3D Mix Interface:** specifically **P3_{C6}** concerning build-all-then-view mentality
- **P5 Selecting and Editing Object:** The WYSIWYG interface combined with the object selection interface resolves this issue. See later for more detail.
- **P7 Interaction specification,** specifically **P7_{C3}** on re-enforcing two-step mentality and **P7_{C6}** on locating and visualising interactively linked objects

6.3 Interaction Subsystem

6.3.1 Background

The interaction subsystem handles actions generated by various entities within the VE, as well as between the authors and the VE (during editing). Often actions generated by one entity can result in reactions by other entities. The subsystem also runs as a bookkeeping system, keeping track of all the actions that have taken place in order to provide rewinding and undo facilities.

Journaling System

One of the major applications of journaling systems is in file systems and databases, where operations are kept in a journal for backtracking, recovery and integrity maintenance processes [84]. Examples of files system offering journaling functionality includes Apple's HFS (although it is turned off by default), Linux Ext3, ReiserFS [82], Solaris UFS and IBM's JFS [8]. Most journaling systems have to balance their main tasks (ensuring structural consistency and recoverability) with robustness, so as not to disrupt actions that the system is trying to track [8].

We employ a pair of journal-linked stacks to keep track of different types of actions. These stacks operate similarly to a journey. One stack is used to track any authoring actions performed by the author (thus allowing for the undo functionality), while the other is used to track VE-based actions (thus allowing for the rewind

functionality). The system writes to different stacks depending on the type of the entity which has generated the action.

Types of Action-generating Entities

There are three types of entities and each generates a different type of action. The different entities are VE-based Objects, the author, and Independent Monitors. Each are described below:

1. **VE-based Objects:** These are the objects inside the virtual environment itself, and as such, any actions they generate are affected by the VE clock. The interaction subsystem keeps records of any VE-based actions in the time-based stack for rewinding purposes.
2. **Author:** These are actions performed by the author and are performed outside the VE clock. The clock is paused during any such action. The subsystem keeps a record of any authoring actions in the authoring stack for undo purposes.
3. **Independent monitors:** An independent monitor reacts to certain conditions differently depending on whether the clock is running or not. An example of an independent monitor is the collision detector which constantly monitors and prevents objects from entering each other, and pushes them apart when they try to. If the VE clock is paused, the pushing apart action is treated as an authoring action⁹ and so is added to the authoring stack. On the other hand, if the clock is running, the pushing apart action is treated as a VE-based action.

Each of the entities, and how they generate actions, is discussed in the following subsections.

6.3.2 Authoring Actions

In order to overcome the multiple tools issue, there is a need to eliminate (or at least minimise) the line between the main tasks. This issue mostly exists in tools which have clear, separate tasks such as environment building and interaction specification. Ideally, there should be no notion of macro-steps such as 'build environment' or 'define interactions', but rather the notion of micro-steps. This would mean that small tasks can be performed in any order as the author sees fit. These micro-tasks should be a reversible step in themselves, and after each task is completed, they should leave the VE in a stable state. Ideally, authoring actions should correspond with the author's smallest possible step in building a VE. In

⁹ Since only authors can manipulate time. There is also a requirement that non-instantaneous action including pushing apart objects, must be based on a VE clock. With the VE clock not running, it is instead based on a real clock but could not be added to the VE-action stack since it can not be rewound (but it can be undone).

essence what need to be avoided is arbitrarily grouping each of these authoring actions into subtasks. This is what has been done with older tools, which basically separates into environmental building and interaction specifying subtasks. Instead, we can provide the author with the ability to perform smaller authoring tasks which are in line with the way authors conceptually create VEs. Figure 92 below presents example subtasks identified by authors who participated in our study. They have been grouped along the line of the subtasks used in most authoring tools.

Microtasks also correspond to atomic operations required by most backtracking and undo systems. Atomic file operations, for example, ensure data integrity and recoverability but making sure that operations either succeed or they do not. In order for this to work and still ensure robustness, these operations have to be small and limited in effect [82].

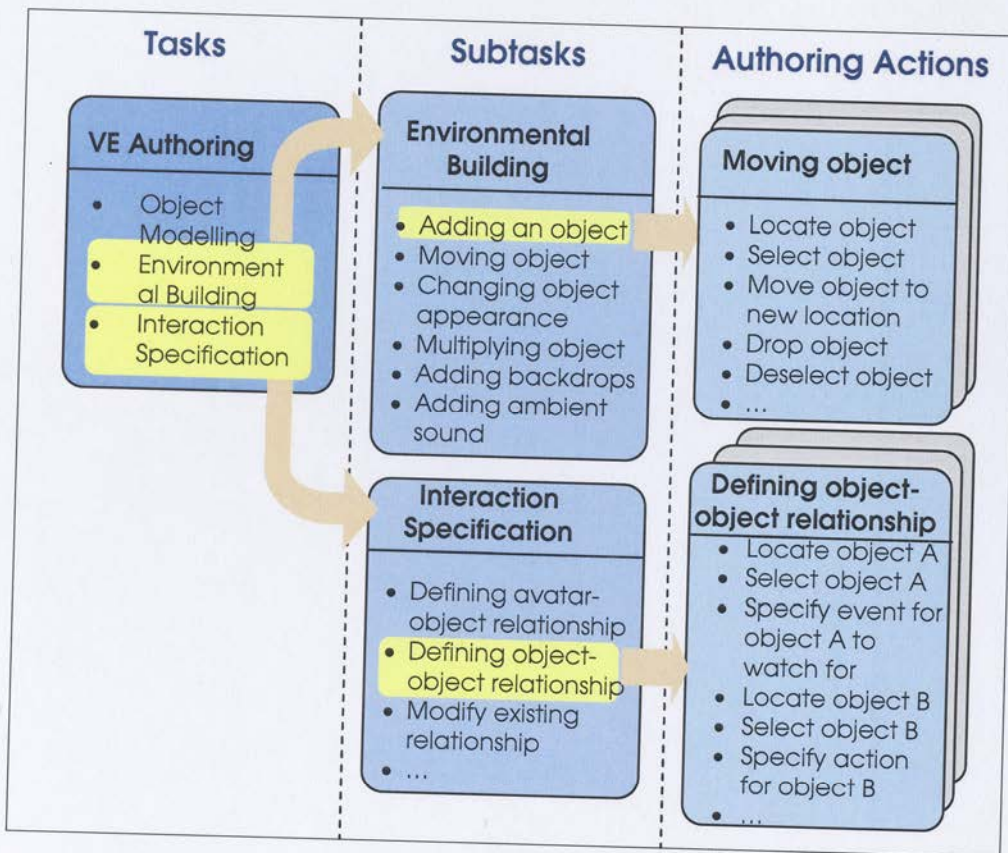


Figure 92: A possible task breakdown for a VE authoring process. The two main tasks of environment building and interaction specification are broken down into smaller subtasks (not all are shown). Some tasks are broken further to even smaller steps or authoring actions

What do Authoring Actions Solve?

- Since authoring actions match closely to the way authors think, initial learning time should be reduced.

- **P1_{C3} Multiple step mentality:** By not forcing particular task groups, authors are afforded the ability to use their own style of authoring. This comes about because the tool does not force any particular combinations or model.
- **P1_{C4} Unnecessary saves or backups:** Each authoring action, being completely compatible with one another and reversible on its own, can be translated directly into an undo block. This contributes towards reducing the need to save unnecessarily. (One of the major reasons given for the high number of saves was the use of multiple tools.)
- **P1_{C2} Incompatibility issues amongst tools:** All tasks can be built on a common group of authoring actions, leading to a single interface for all tasks. There is no incompatibility problem and no switching between tools or modes.

6.3.3 VE-Based Actions

VE-based actions are actions that can occur within the bounds of a virtual environment and, if time-based, are dependent on the VE clock. All VE-based actions are object-based, in that they are *performed* by an object in the VE. The World object (see later) is used to perform abstract actions that may not be associated with a particular object. The avatar object is used as proxy to the user generated actions, while the VE clock itself is used to generate time-based actions. Note the difference between avatar-based action that occurs within the bounds of the VE and authoring actions that occur outside these bounds.

VE-based actions can be further subdivided into the following subtypes

1. **Informative (sometimes referred to as event-based) Actions** (generated by objects in the watcher role): These actions are used to inform other objects of the event generated or experienced by this object. They happen instantaneously¹⁰.

Informative Action	
Type	Type of informative action, such as collision, avatar input action etc.
Time index	The time index that this action was initiated
Generating Object	Object that initiated this action
Target Object	Object to which this information was sent (each recipient will generate one information unit).
Additional Information	This is dependent on the type, for example, the collision type will contain information about other object in the collision.

¹⁰ Instantaneous action starts and finishes within the same instant.

2. **Performance Actions** (generated by a doer role): These actions are performed in response to one, or a group of, event generating actions. They are either time-based or happen instantaneously.

Performance Action	
Type	Type of performance action, such as moving an object etc.
Time index	The time index that this action was initiated
Generating Object	Object that initiated this action; usually this is the object that is actually doing the action.
Additional Information	This is dependent on the type, for example, the collision type will contain information about other object in the collision.
Estimated Duration	This is the estimated time for this action to complete. This is used to determine when an action is completed.
Allow Interruption	This is a flag to determine whether or not this action can be interrupted by other conflicting action. The default is yes.
Repeat	This is a flag to determine whether or not the action should be repeated. Repeated actions are usually stopped by explicitly or via interruptions.
Action Group	This field is used by to store information about the nature of the action. The system uses this information to keep determine action conflict and interruptions.

From the point of view of the authors, each object in the world acts independently of one another and the world. An object can generate different types of informative actions, for example, collision with other objects. An object can also react to informative actions by doing a performance action or generate further informative actions. Figure 93 shows different types of actions in a simple example.

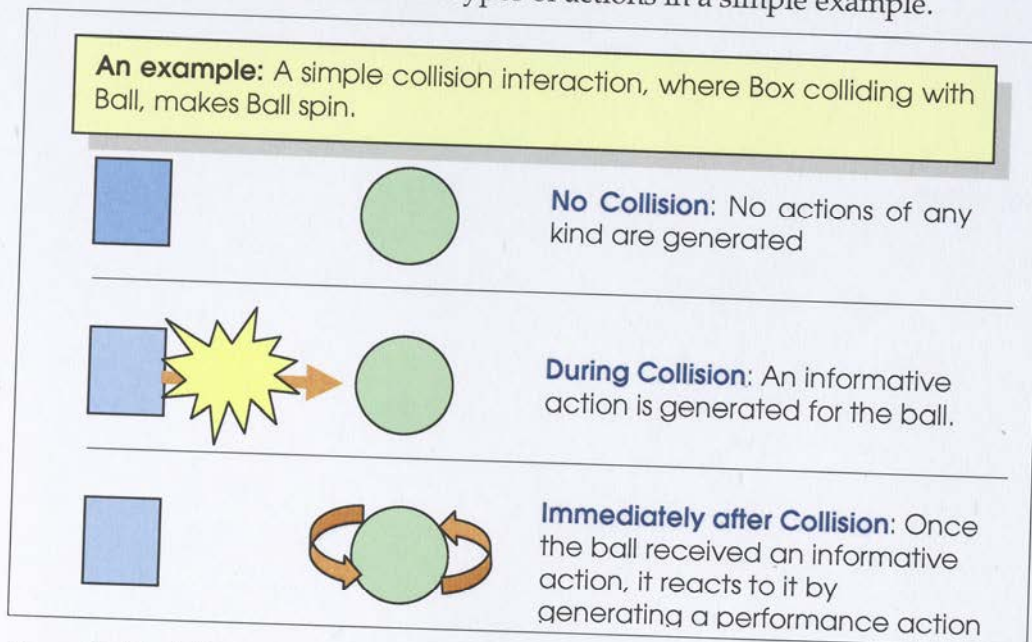


Figure 93: The different type of VE-based actions that were generated during a simple interaction.

What do VE-based Actions Solve?

- **P8_{C4} Pause, Rewind, Fast-forward Functionalities:** The system's action roll-back mechanism enables the rewind features to work. There is however a point to remember that fast-forwarding immediately after a rewind will not restore the state before the rewind procedure.
- **P6_{C2} Minimal object mapping & P7_{C2} Avoidance of complex interactions:** The system enables the author to think of objects in the environment as independent and this could simplify interaction specification. Object-based specification should allow for easier understanding.

6.3.4 Monitor-based Actions

Independent monitors are based on the idea of intelligent agents. For example, interface agents actively monitor user's actions and *intelligently* apply changes that it thinks the user wants. Their main purpose is to reduce work and information overload to the user [39]. They have been described as the opposite to direct manipulation [59][40], in the sense that they act indirectly and that the results might not be what the user wants or expects. However, it has been suggested that it is a better route to take if users are not well trained in the software. Another suggestion is that software should adapt to users (this is what a good interface agent should do) and not the other way round [40].

We make use of agents to check for certain conditions and to react by simulating other actions. We call them independent monitors because they operate independently from the interaction subsystem (just as the author and the end-user are), and constantly check for and react to the condition being monitored. Examples of monitors are the gravity simulator and collision detector.

The main reason for using an independent monitor is to ensure common effects whether it is as a result of an authoring action or a VE-based action. For example, authors would like the gravity effect to occur whether the clock is paused or not. If the clock is paused, the object falls with respect to the real clock. Furthermore, it makes sense for this reaction to be part of an authoring action set rather than a VE-based one. However, if the clock is running, the falling object reactions are recorded in the time-based stack. Furthermore, every object should be able to opt out of the monitoring services by means of setting a flag.

Essential Monitors:

- **Gravity Simulator:** The gravity simulator monitors applicable objects for when they are floating. In our system the simulator displaces the object's position at a predetermined rate periodically in relation to the appropriate clock.

- **Collision Detector:** This detector monitors for when objects are entering each other's boundary and prevents that from happening. Collision detection is a very active research area. It is very important in simulating a believable environment and has been shown that there are complex factors involved such as velocity and rebound force; a small change in this factor could cause the simulation to seem less believable [48]. In our interview sessions, we also found that our authors found that quality of collision detection to be very important to the realism and believability of their environments¹¹ as well as useful during their authoring.

We implemented a simple bounding-box-based collision detection algorithm for demonstration purposes. Advanced algorithms may be used instead to provide a more accurate and efficient detection scheme [34] including more recent hardware-based approaches [32] [25].

What do Monitor-based Actions Solve?

- **P4_{C3} Difficult in placing objects:** With the WYSIWYG interface and two available viewpoints (top-down and first person), the introduction of gravity and collision detector allows authors to move and place objects with sufficient accuracy.
- **P6_{C1} Lack of gravity and Collision Detection:** One of the main usages of independent monitors is to simulate real world conditions such as gravity and collision detection, and do it in the same way whether or not it is as an authoring action or a VE-based action. This allows authors to exploit their knowledge of the real world in their authoring process.

6.3.5 Interaction Manager, Undo and Time Handlers

The three components of the interaction subsystem are as follows:-

1. **Interaction Manager** handles all actions and interactions, both generated from the real world and within the virtual environment. It does this with the authoring action handler and the VE-based action handler respectively. The Interaction Manager also makes decision on potentially conflicting actions on a latest-override-previous basis (though this can be modified to another conflict resolution policy). This means that whenever a new action is requested, any current action involving the objects in the new action halts and the new action is started. An example of how this works is given in the next section on *what does this solve?*

¹¹ For those authors for which realism and believability are one of the main aims of their virtual environment.

2. **Time Handler** manages the rewinding facility and it does this by back-tracking the VE-based Action Stack (VEAS). The VE-based action handler writes a log entry for every VE-based action to the VEAJ for this purpose.
3. **Undo Handler** manages the rewinding facility and it does this by back-tracking the Authoring Action Stack (AAS). The authoring action handler writes a log entry for every authoring action to the AAJ for this purpose.

The Interaction Manager, together with the Time and Undo Handlers, is responsible for handling all the actions generated by all objects in the environment. It is also responsible for undoing all actions in cases when authors or consumers want to rewind the environment. (Fast forwarding does not involve the system because when you fast forward, the clock is sped up). In order to back track the interaction subsystem stores the following information unit for each action.

The system also stores other information necessary for back-tracking including end of performance action¹² and interruption information¹³.

What do Interaction Manager, Undo and Time Handlers Solve?

- **P8_{C4} Pause, rewind, fast-forward functionalities:** Time handler and VEAS allows for rewind to be possible.
- **P1_{C4} Unnecessary saves or backups:** Undo handler and AAS allows for undo functionality.
- **P7_{C1} Tools require extensive programming skill & P7_{C2} Avoidance of complex interactions:** Instead of specifically handle all actions; authors can leave it to the system. For example, in a situation where an actor is to say different thing based on different conditions:

Condition 1: Say 'That is a lovely dress ... ' [talk about dress]

Condition 2: Say 'This is a lovely day ... '[talks about weather]

If the author was to handle everything the interaction might have to be specified as:

¹² End of Performance Action Information: Specify when a performance action is ended. By default, performance actions are not started again if the event it is listened for is activated. An end-of-performance message signifies that the actions can be performed again.

¹³ Interruption Information: Some actions belonging to an object are interruptible. During the course of an action performance, another event might occur and ask the same object to perform another action which is in conflict with the current one. This interruption needs to be recorded for roll-back purpose.

```

IF (Condition1 && NOT( ISDOING( talk_about_weather) ) )
    DO( talk_about_dress )
IF (Condition1 && ISDOING( talk_about_weather ) )
    STOP(talk_about_weather )
    DO( talk_about_dress )
IF (Condition2 && NOT( ISDOING( talk_about_dress ) ) )
    DO( talk_about_weather)
IF (Condition2 && ISDOING(talk_about_dress ) )
    STOP(talk_about_dress)
    DO( talk_about_weather)

```

By letting the system keep track of actions and handle interruptions, authors might be able to specify interactions with just:

```

IF (Condition1 )
    DO( talk_about_dress )
IF (Condition2 )
    DO( talk_about_weather )

```

6.3.6 Interaction Subsystem Overview

Figure 94 illustrates the different components that are linked to, as well as those that are part of the interaction subsystem. In the rest of this section we trace the route of the most common tasks through this overview diagram. To do this, we make use of the numbered arrows in Figure 94, we refer to them in square brackets [].

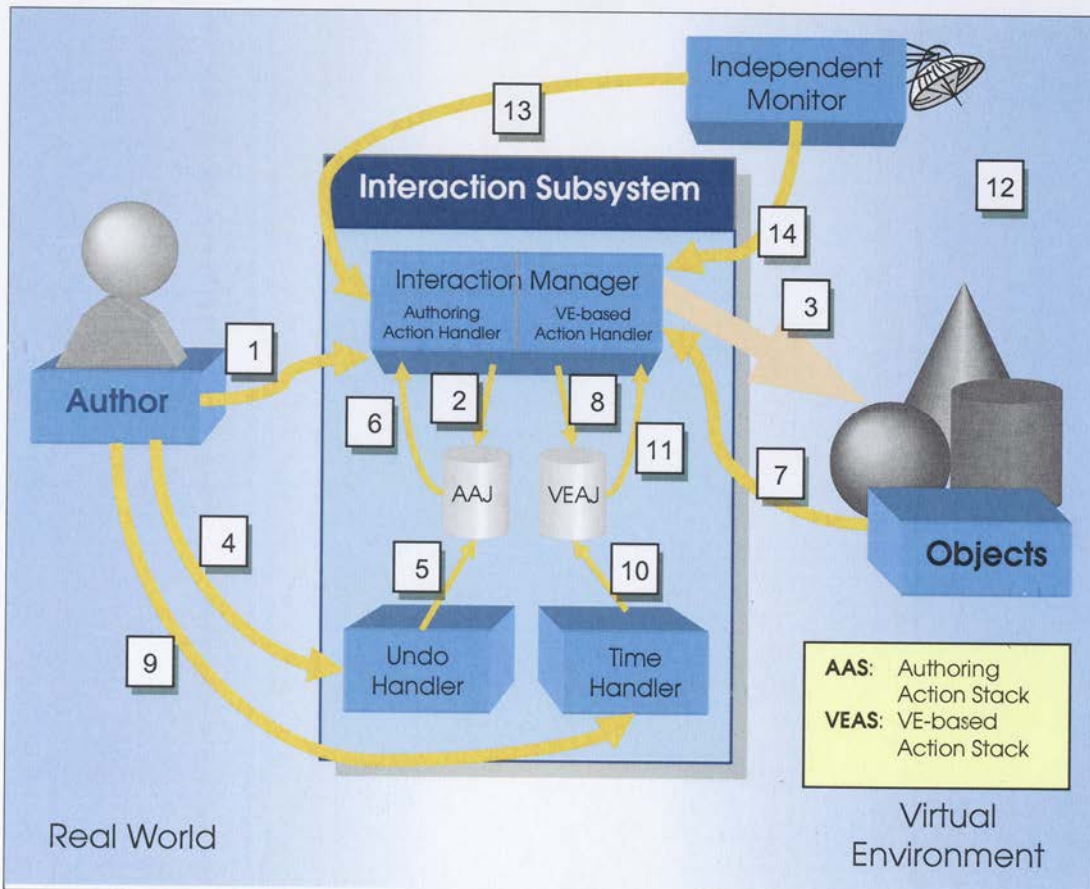


Figure 94: Visual overview of the interaction subsystem

Author performs an authoring action

- [1] Author's authoring action is sent to the interaction manager. If it determines the requested action can be performed, the next steps are followed. Otherwise the process stops at this point.
- [2] The interaction manager adds an entry into the authoring stack .
- [3] Interaction manager affects the requested change to the virtual environment.

Author undoes an authoring action

- [4] Undo request sent to the Undo handler
- [5] Undo handler requests the reverse of the last action in the stack
- [3] Interaction manager affects requested change

An object sends out an informative VE-based action

- [7] The informative VE-based action is sent to the interaction manager. If it determines the requested action can be performed, the next steps are followed. Otherwise the process stops at this point.
- [8] The interaction manager adds an entry to the timed action stack .
- [3] Interaction manager sends action on to the target object.

An Author (or consumer) rewinds the virtual environment

- [9] Author sends rewind request to the time handler.
 - [10] Time handler requests the reverse of the last action in the stack
 - [3] Interaction manager affects the requested change
- Repeat until required time index is reached.

An independent monitor picked up a matched condition

- [12] A monitor picks up a matched condition and will attempt to effect the required change.
- [13] If the clock is not paused, the monitor requests the change as an authoring action, where the usual authoring action processing takes place.
- [14] If the clock is paused, the monitor requests the change as a VE-based action, where the usual VE-based action processing takes place.

6.4 In-Environment Interface

6.4.1 Introduction

In-Environment interfaces require that any authoring interface appears within the virtual environment itself.

What does In-Environment Interface Solve?

A number of in-environment interfaces are presented in this section and each resolves different issues. Here we present issues which the in-environment interfaces, in general, can resolve. Each of the individual interfaces will also present its own resolutions.

- **P3_{C2} Constant switching between mindset:** By making all previously external (and often 2D) interfaces appear inside the environment, the authors do not have to switch mindset.
- **P3_{C3} Cluttered interface & P3_{C4} Reduced viewing area:** As well as reducing interface cluttering with the authoring interface hidden, the 3D portion is increase to fill up almost all the available space. Additionally, authors are in the exact same position as consumers when not authoring. This lets them switch roles between author and tester with minimal effort.
- **P2_{C2} Lack of immediate feedback:** Any changes are applied immediately on leaving the authoring interface, allowing authors to see the changes straight away without a build delay.

6.4.2 3D Cursor

This is a representation in the environment that the author or consumer can use to point at, select or interact with objects in the environment (see Figure 95). A Cursor is mainly used to select a three dimensional point or an object in the VE. Object selection is dealt with in the next subsection.

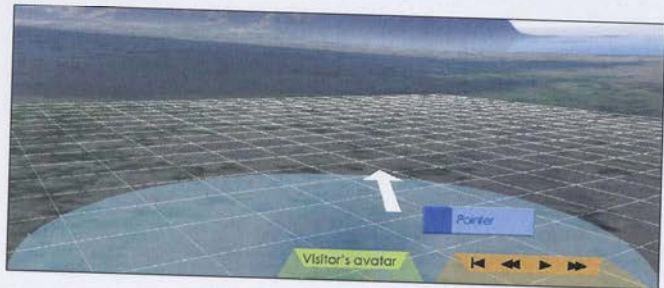


Figure 95: Screen Capture from VEBUILDER I, showing the 3D pointer and its reach. The pointer is manipulated using the mouse while the avatar's movement is manipulated using the keyboard.

Accuracy versus Ease of use

Accurate positional indicators in a 3D environment using a 2D based display and locator device system can be very complex [47]. Studies have been made into factors which might make it easier for users to select a 3D position sufficiently accurately. This includes using transparent volume pointers [90], adding shadows [33] and varying the pointer size with objects within close proximity [27].

It was felt by most authors that the plain volume pointer is more difficult to judge and there is a potential for confusion as many objects may be inside the volume. We decided that it was best to use a plain point-based pointing system which the author can think of as an extension of an avatar's arm and hand. This was implemented in

our system's first-person viewpoint. In this view, the author is able to use the mouse to move the 3D pointer as though it is an extension of the arm of the invisible avatar. The mouse roller can be used to manipulate the pitch of the pointer. Furthermore, we also incorporate a shadow into the pointer to help authors and end-user judge distances from the ground as suggested by Hudson [33]. We envisage a more sophisticated version of this interface to include a virtual arm and finger instead of a pointer.

From the bird's eye view (accessible only to authors), authors are able to access all objects within the view screen below the current height of the pointer. This height can be adjusted using the mouse roller.

Context Preservation and Consistency

The 3D pointer animates and moves into a different orientation and position to facilitate different tasks. The author never loses sight of the pointer and the same pointer is used to access all interfaces within the virtual environment. In fact, it is also the same pointer that the consumer would see when viewing the virtual environment.

What does this solve?

- **P3_{C1} Pointer Confusion:** This solution unifies the different pointers. The smooth transition between the two viewpoints makes sure that authors do not lose the pointer.
- **P4 Multiple viewpoints object placement interface:** The 3D pointer is also helpful in placing objects in the environment by providing sufficient accuracy in both first-person and bird's eye viewpoints

6.4.3 Object Placement and Selection Interface

Selecting Objects in 3D

Most VE object selection methods are based either on the touch or point system. Touch system is where the object which is touched last by the cursor is the one selected. Most work via the collision detection process whereby the object that was last collided with the cursor is the selected object (until the second collision is registered on the way out then the object is unselected). A simpler version of the touch is the bounding-box or volume based selection where each object's bounding box is testing against the cursor (or its bounding box). An example of this is the Go-Go interaction technique [53] where the cursor is represented by a hand and this hand can be extended to reach far-away objects.

The other selection method is based on ray-casting or image plane-based interaction techniques [52]. Often this results in the user pointing at an object and whichever object lying directly underneath the cursor (on the 2D projection that 3D objects in the scene make on the image plane) is the one selected. Ray casting is sometimes used to achieve similar results.

Variations of the two techniques were compared in [10] with results indicating that different each one is suitable for different situations. For example, the touch based system is slower for selecting faraway object than the point system, while the touch system is more suitable for selecting objects that are in a cluster.

In our system, we use the touch system for object selection while in first person mode. This is because our arms-reach models means that selectable objects will be close-by, making it more suitable for the touch system. Additionally, this is also in keeping with the arm metaphor as authors can reach out and touch the object to select it. This is also in keeping with allowing the author to act the same way the end-user controlling an avatar is able to do.

However, since the objects are further away in the bird's eye view mode, we use a better suited system based on image plane-based interaction techniques whereby the closest object directly underneath the cursor is selected. Different layers of objects can be selected by using the roller on the mouse.

Placing Objects after Selection

Object placement can be done from either first-person or bird's eye viewpoints. The 3D pointer points downwards to indicate the target position and a rotating red cross also appears to facilitate accurate placement (see Figure 96). With the aid of gravity and collision detection, a dropped object automatically falls to the ground and moves away from any object already occupying the same space.

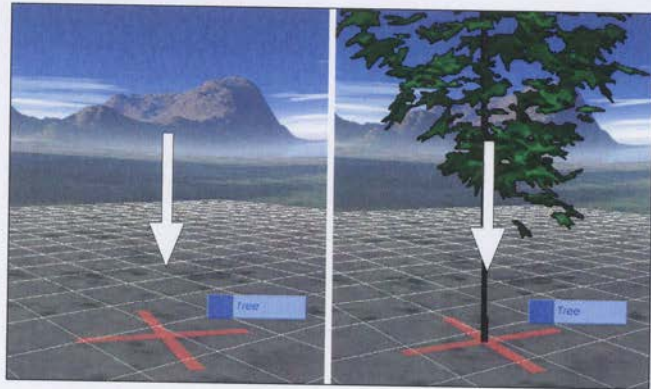


Figure 96: Screen captures from VEBuilders I, showing how new object can be placed inside a virtual environment. When placing objects, the 3D pointer points downwards to indicate the target position. A rotating red cross also appears.

Interacting with Objects

At any time, an author can hold down the primary button to activate a context-based revolving menu (see next point). A single click-release will be treated by the system in the same way as the end-user would (in our system, this generates an 'avatar input' informative action).

What does this solve?

- **P4 Multiple viewpoints object placement interface:** Together with two different viewpoints available, gravity and collision detection systems¹⁴ in place, as well as default object placement behaviour¹⁵, authors can easily and accurately place objects in the virtual environment.
- **P3_{C1} Pointer Confusion:** There is only one 3D-based cursor which is used to access both the authoring interface or interact with objects the same way an end-user would.
- **P2_{C1} Different representation & P2_{C4} Different Control for edit and view modes:** There is only one way of interacting with objects for both authors and end-users. See next section for more detail.
- **P5 Selecting and Editing Objects:** Objects can be directly selected using the cursor in both viewpoints. This should minimise the confusion and complexity usually involved with selection. It should also be easier to use and authors can keep track of the selected object easier since in addition to being highlighted, selected objects are always visible in the view area in both first person (within arm's reach) and the bird's eye view (within the view pane).

6.4.4 Revolving Menu

2D Menu

Most window systems employ cascading rectangular menus to display different functionalities to the user. There have been other types of menu such as pie, where every entry is placed the same distance away from the point of origin. There have been a few comparative studies of the different types of menu display including [13]. Both methods suffer from cluttering as more items fill up the available area quite quickly. Pie menus are particularly affected as the pie has to grow bigger in order to fit in menu texts, increasing the distance further. Bederson [7] proposed the fisheye menu in which distant entries are smaller depending on their distance from the current item.

There are also a few ways to select a menu entry with the click being the most obvious and popular case. Some systems including Microsoft Windows automatically expand a submenu after its parents have been highlighted for some time. A system [36] based on the cursor direction (a submenu expands when the user moves the cursor horizontally over its parent) was found to not be significantly faster than the traditional method. However, the author did suggest that since menu usage improved with time, his method might still prove to be faster.

¹⁴ See 6.6.3 Special Objects: The Avatar and World Objects for more detail

¹⁵ See 6.6.1 Objects for more detail.

It was decided that menu selecting would be done with a single click, with automatic selection after 3 seconds. This was found to be familiar to most authors.

Activating the context menu

Activating a context menu is often done using a dedicated mouse button (the right click for a right-handed user in most windows system) will display a menu with entries dependent on the location of the cursor at the time. Recently, there has been another activation method, used mostly with a pen-based input – the click and hold method. The context menu is activated when the user hold down the pen until a certain time is reached¹⁶.

After activation, a rectangular menu usually appears with one of its corners at the cursor point; usually the top-left. This is different if there is not enough space; with the menu appearing at places where sufficient space is available. A pie menu appears with the center at the cursor point. There have also been some recent studies [31] on menu location in pen-based devices since the traditional display position of bottom-right is occluded by the pen-holding hand.

3D Menu

Due to the requirements that the cursor single button press or mouse click be used to interact with objects in the same way an end-user would, we could not use the traditional right-clicking method to activate the context menu. To get around this, we took the idea from the pen-input based devices where holding down a mouse button activates the context menu.

We chose a combination of pie and rectangular menu display techniques, resulting in the wrap-around circular menu display (Figure 97) to overcome the potential for cluttering up the screen as well as achieving a fisheye menu effect. We were also able to avoid having the cursor changing mode into menu selection mode by letting the user move between the entries by using the mouse wheel to rotate the menu. The currently selected entry is the central highlighted item. This method fits well with the roller and allow for direct mapping between the input device and the interface.

A single left click select the entry, while a single right click will retract the current menu and moves up a menu level. Holding down right click will retract the menu all the way back and cancel the menu selection. After entry

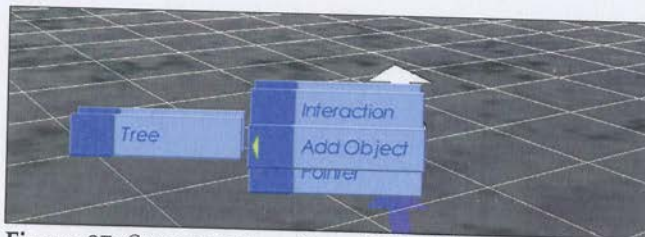


Figure 97: Screen capture from VEBuilders I, showing the 3D revolving menu interface.

¹⁶ This is accompanied by dots appearing around the contact point with each dot changing colour to indicate passing time. The context menu is activated once all the dots have changed colour.

completion or cancellation the clock reverts back to its state before menu display, either resuming or carrying on in pause mode (if it was in this state before). Holding down the key to recall the menu means that authors are able to interact with the environment in exactly the same way as an end-user would do.

Once the menu is displayed, the VE clock freezes (if the clock is not in pause mode already), effectively pausing the simulation.

What does this solve?

- **P2_{C1} Different representation & P2_{C4} Different Control for edit and view modes:** The menu system runs in addition to the end-user control system, allowing authors to operate in the VE in the same way as the end-user would.
- **P3_{C1} Pointer Confusion:** The pointer is not used in the selection of menu entries, leaving the pointer with only one function – to interact with the objects in the VE.
- **P3_{C2} Constant switching between mindsets:** With the menu appearing within the environment and the pointers not involved in menu operation, there is no transfer of the pointer between 2D and 3D representations.
- **P8 Time Manipulation:** Time freezes while authoring is done, allowing authors to make changes to the environment.

6.4.5 In-Environment Console Bar

This is the 3D equivalent of the toolbar and status bar. The console bar is located at the bottom of the 3D window and has an appearance of being within the virtual environment to avoid the two-dimensional type of interface that is usual for a status bar or a console. To access the console bar, authors move the 3D pointer towards the screen where the pointer will rise over the console and allow the icon to be pressed. Authors can also drop a selected object on to the icon. Amongst the useful icons on the toolbar are the time-control interfaces (only available during authoring), the world and avatar representations and the object Library¹⁷. It also acts as a status bar giving information such as current time index and information about the highlighted or selected object.

The console bar gives authors access to other functionalities as well as displaying other relevant information such as current time index, in a 3D-based interface. This was done to make sure that the interface for displaying status information would not cause further problems concerning 2D/3D mixed interface.

¹⁷ See 6.6 Objects and the Object Library for more detail.

6.4.6 Speech Bubble Interface

The speech bubble interface's primary aim was to provide a way for authors to edit an object's attributes or to modify its behaviour. The problems with the usual attribute in a window or a dialog box, (as pointed out by authors in comment P5_{c1}) was that there is no clear link between the object being editing and the interface. However, there have been similar linking problems within existing operating systems and a good solution has been developed to fix it. The balloon help in Mac OS (and also recently in Windows XP see Figure 98), links any information box with the object that the information applies to.

Not only would a speech bubble type interface solve our linkage issue, but it also allows the interface to appear to exist within the 3D environment. In our system, a speech bubble like interface (see Figure 99) is displayed, pointing back to the associated object, when an object or an interaction involving that object is being created or edited.

Furthermore, there is a smooth transition between the 3D pointer to the more 2D pointer that can be used manipulate the 2D interface in a similar way to the usual dialog boxes. More information about the interaction specification interface can be found later in the chapter.

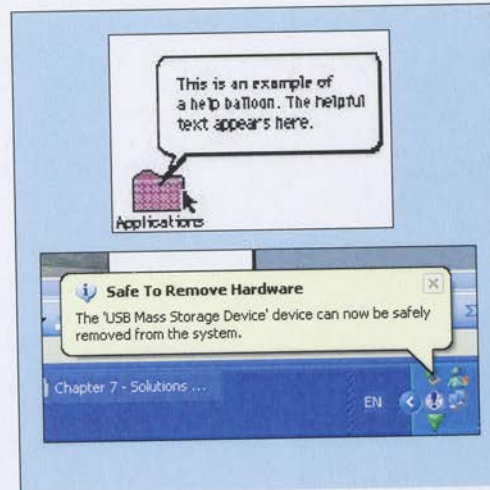


Figure 98: Balloon help interface from Mac OS (top) and Microsoft's version from Windows XP (bottom)



Figure 99: Screen capture from VEBuilders III, showing the speech bubble interface. This interface can be used to manipulate objects or create an interaction.

What does this solve?

- **P1 Multiple Tools:** The object-based speech bubble interface allows the author to focus on the object in order to both edit attributes and add interactions¹⁸ involving the said object. This means that all tasks associated with the object can be done via the same interface.

¹⁸ See 6.5 Object-based Interaction Specification Interface for more detail.

- **P3_{C7} Difficulties in directly editing an object's attributes:** The 3D pointer moves up to the speech bubble smoothly allowing authors to edit or manipulate the selected object.
- **P5_{C1} Confusing Interface:** Authors were experiencing difficulty with some attribute editing interfaces because they force them to have to remember which of the objects is currently being edited. The speech bubble interface also indicates the selected object (in addition to the object being highlighted) with the *tail* of the speech bubble.
- **P5_{C3} Keeping track of selected object:** Selected objects are linked to the editing interface directly by the speech bubble. Authors do not have to mentally make this association themselves.
- **P7_{C6} Locating and visualising interactively-linked objects:** The speech bubble interface is also employed in specifying interactions. See 6.5.1 Visual Specification Interface for more detail.

6.5 Object-based Interaction Specification Interface

6.5.1 Visual Specification Interface

Visual Programming

At the heart of most VE authoring tools' interaction specification interface is the use of programming languages. Yet programming languages themselves have faced issue similar to what this thesis is exploring, namely their unfriendliness towards people who have no special computer training. Visual programming was proposed as a mean of lowering the level of computer skills required to program [61]. It has been described as applying *direct manipulation* [59] to the Graphical User Interface itself [9]. However, early direct manipulation to GUI lacks the means of abstraction and can not be used to program general correspondences between application data and its visual representation in the GUI [46]. Miyashita et. al. also proposed the concept of declarative programming by visual examples. Declarative visual programming languages allow programmer to define behaviours of GUI elements.

Object-orientated (OO) programming languages also lend themselves well to the visual programming paradigm [12], resulting in the development of many OO visual programming languages and related programming tools. Most of these languages tend to only target a specific problem area such as modelling, simulation and process control [44]. Examples of these are Stella [78], and LabView [80]. At the other end of the spectrum are visual languages developed specifically for students in order to teach them about programming. Examples include SIVIL [41] which has been found to aid or speed up the learning curve for students.

Our Interaction Specification

We employed a declarative object-based interactive specification system, borrowing upon useful principles from visual OO programming languages and related tools. Together with the direct object selection and speech bubble interfaces outlined previously, a powerful object-based interaction specification system can be formed.

In our system (see Figure 100 for screenshots), the two parties involved in an interaction are watchers and doers (watchers watch for certain conditions and then send an informative action message to the doers, who will in turn perform the required performance action). There is no specific order in which objects should be assigned these roles. They do not even have to be different objects; one object can act as both watcher and doer. Note that, in order for this interface to work properly, authors need access to the advanced object library¹⁹.

Simply put, an author can select an object and specify its behaviour. For example, she can elect it to watch for certain events and do something in reaction to that event, including telling other objects to do something else. In our system the author can do this as follows:-

She can use the 3D pointer to highlight the object and then use the revolving menu to elect it as a watcher (or doer in an interaction). At this point the speech bubble interface for this object would appear, displaying different options for the role of a watcher (or doer). Once the role is specified, the 3D pointer shows a *chain link* icon and the recently assigned object shows a watcher (or doer) icon; both of which signify that an interaction is being specified. Authors can move on to another object to assign it as another watcher (whose watched event can be logically combined to existing ones with an *and* or an *or*) or a doer. A similar speech bubble then appears to complete the specification. After all the

objects have been assigned authors can access the revolving menu and select *complete interaction* to finish the specification process. All these linked objects are said to be in an interaction chain. A list of all interaction chains specified is accessible from a button on the console bar.

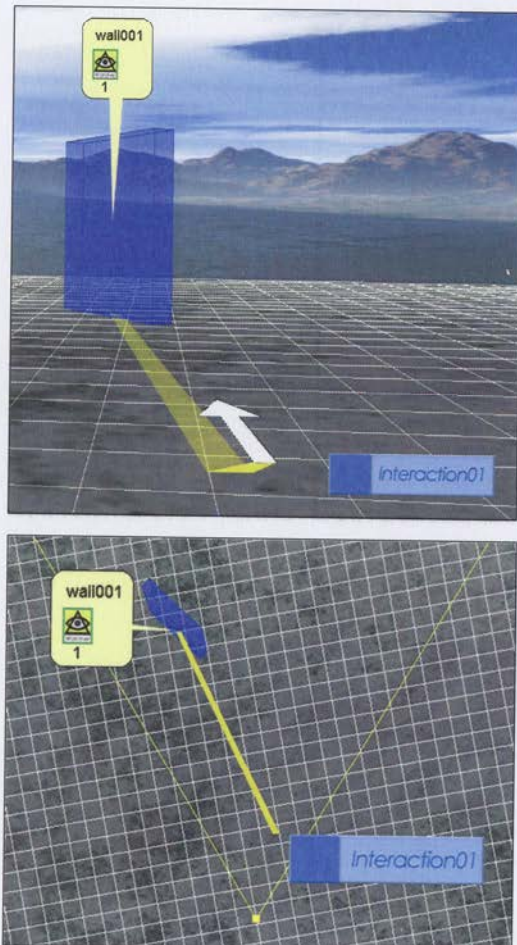


Figure 100: Screen Capture from VEBuilders III, showing the author in the middle of specifying an interaction. The wall has already been selected as a watcher. The yellow line indicates an interaction link. The author must now select and configure another object to complete the specification. First person view (top) and top view (bottom)

¹⁹ See 6.6.2 Object Library for more detail

Additionally each object also lists any interaction chains that it takes part in separated into those for which it acts as a watcher and those for which it acts as doer. Additionally, whenever an interaction chain is being created or viewed, all objects will have mini-speech bubble icons over the head and a yellow line linking all of them.

6.5.2 Advanced Interface

Advanced authors should also be able to quickly specify interactions via an advanced interface that lets the author select objects from a list, so that time is not wasted looking for objects. Additionally, a programming or scripting language like interface should also be provided for an even faster way of interaction specification.

6.5.3 What does this solves?

- **P7_{C1} Tools require extensive programming skills:** The visual specification interface allow for authors to specify interactions without having to learn how to program. The more advanced interface allows for experienced authors to specify interactions in the way that they are familiar with.
- **P7_{C6} Locating and visualising interactively-linked Objects:** Interactively-linked objects can be visualised in a chain (or web in cases of multiple objects).

6.6 *Objects and the Object Library*

6.6.1 Objects

Authors need to populate virtual environments with 3D models and add interactions between them. However, as mentioned in the previous chapter, most authoring tools do not take advantage of the authors' knowledge of existing real-world objects. In the interviews, most authors have indicated that most of the objects in their environments are based in the real world. It makes sense then to provide them with templates of objects that look and act like their counterparts in the real world. Authors can then add an object by choosing from a range of templates available in the library.

In our system we provide objects that are of the following characteristics:

- Each object has a 3D model in the standard ratio; in our system we used 1 VE unit to 1 meter.

- If applicable or available, changeable attributes such as shirt colour, hair colour, etc.
- Possible informative actions – these usually include collision, avatar select, proximity sensor, etc.
- Possible performance actions – For examples actors are able to do animation, speak. A ball might be able to bounce, etc.
- Default behaviour – Some objects may have default behaviour built-in. For example, a tree in our system has a 'bill-board' action built in. Some objects maybe have default placement behaviour as well. For example, a wall in our system automatically links to the nearest wall to form one single object.

The object should reflect its real world counterpart (if applicable) as much as possible, both in appearance and behaviour.

What does this solve?

- **P6_{C2} Minimal object mapping & P7_{C1} Tools require extensive programming skill:** Making an object act and react the same way as its counterpart in the real world allows authors to easily think about the interaction. They do not have to think about how they would translate what they think into code, but rather they can translate it to something closer to what they have in their heads.

6.6.2 Object Library

Objects are made accessible to authors via the object library. It is available from both the revolving menu and the icon on the console bar. The object library allows authors to add an instance of an object in the library to the virtual environment.

6.6.3 Special Objects: The Avatar and World Objects

In order to fully use the object-based interaction specification system presented previously, authors must be able to select the avatar (representation of the end-user) and other world related attributes such as weather, time of day etc. In our system, this can be done with the Avatar (Figure 101) and World objects. These objects are also accessible via the console bar.

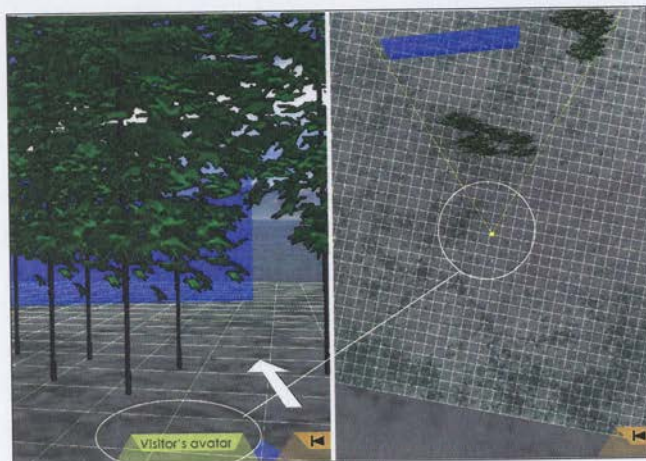


Figure 101: Screen capture from VEBuilders I showing the avatar representation in first person view (left) and top view (right).

6.6.4 The Avatar Object

The Avatar object is used to access the consumer's representation while in the VE. This object also has a model associated with it, and thus can perform any associated animation. The avatar object can also contain extra attributes such as name, health level, location as well as flags to indicate various avatar related conditions.

What does this solve?

- **P7_{C1} Tools require extensive programming skill:** The avatar object provides a way for authors to think about interaction that involves the consumer. It also represents the author in the virtual environment.
- **P6_{C2} Minimal object mapping:** The avatar object allows authors to associate consumer attributes such as name, age etc.

6.6.5 The World Object

The world object is used to contain the following information amongst others:

1. **Time of day:** This information is derived from the current time index. The author can determine what time the starting time should be.
2. **Weather and other environmental information:** The current environmental condition such as weather and time are contained here. The virtual environment is simulated with this condition in mind. This means that the entire virtual environment can be changed via an interaction chain. For example, at a certain time index, the environment could get dark.
3. **Physics condition:** Authors can turn on or off physics-based variables in the virtual environment via the World object. An example of this could be gravity or collision detection. These two physics effects are on by default and thus authors would not be able to place objects through each other. Objects placed in the air automatically fall to the ground or onto other objects (if they are in the way). These physics variables can be turned on or off on an individual object basis. Objects not obeying gravity will be suspended in midair, while objects not obeying collision detection will be able to go through other objects.
4. **Other attributes:** Authors can arbitrarily add any additional attributes to the world object.
5. **God-like Actions:** Actions that cannot be associated with existing objects, such as creating objects on the fly can be done with the world object.

What does this solve?

- **P6_{C1} Lack of gravity & collision detections & P6_{C3} Non-existent global environment specification:** Gravity and collision detections and other forces and phenomena (such as weather or location) are provided by the system and can be manipulated by the author via the world object.
- **P6_{C2} Minimal object mapping:** The world object also acts as a container object for other abstract concepts or attributes that cannot be logically associated with other objects. An example of this could be the current season of the virtual environment.
- **P7_{C1} Tools require extensive programming skill:** The world object can be thought of as a 'God' object that knows of everything and can perform actions not associated with existing objects such as creating an object.

6.7 Resolution Strategies

The solutions outlined in the previous sections contribute to solving the eight interface issues. Each of the interface issues is resolved by a combination of different aspects from these solutions. In this section we present the resolution strategies for each of the interface issues. We do this with the aid of diagrams.

On the following pages, we go through the interface issues outlined in the previous chapter and link them to the solutions mentioned earlier in this chapter. Each subsection starts off with a diagram linking the problem to the solutions that together addresses the problem. This link is indicated by thick grey lines. Another type of link indicated by thin black lines represents the solutions that address a particular comment put across by the authors in the interviews.

6.7.1 Multiple Tools

Resolution Strategy

P1_{S1} Unifying interfaces: All interfaces are accessible and presented in a similar way with the same look and feel. This applies to all tasks and is achieved through the *unified WYSIWYG interface* that allows most tasks to be accessed via the *revolving menu*. For object-based manipulation (both in terms of appearance and interaction), this is done via the *speech bubble interface*.

P1_{S2} Tasks Breakdown: The main problem with the multiple tools is that the different types of tools are fragmented along the so-called main tasks: modelling, environmental building and interaction specification. By breaking down these main tasks into the smallest possible and ensure compatibility between these tasks (compatibility in the sense that doing one tasks does not overlap or negate others), authors are freed from having to work along these main tasks, and can do any subtasks (we call this *authoring actions*) in any order.

Interface Issue

P1 Multiple Tools

A number of tools may be used in VE authoring process, usually one for building environment and one for specifying interactions.

P1 _{C1}	Confusion about different tools
P1 _{C2}	Incompatibility issues amongst tools
P1 _{C3}	Multiple steps mentality
P1 _{C4}	Unnecessary saves or backups

Solution Set

VE clock (6.2)	
Unified WYSIWYG (6.2.2)	P1 _{S1}
Actions & Interactions (6.3)	P1 _{S2}
In-environment interface (6.4)	P1 _{S1}
3D Pointer (6.4.2)	
Object Select & Place (6.4.3)	
Revolving Menu (6.4.4)	P1 _{S1}
Console Bar (6.4.5)	
Speech Bubble (6.4.6)	P1 _{S1}
Obj-based Interaction Sp (6.5)	
Objects & Object Library (6.6)	
World & Avatar Objects (6.6.3)	

 The issue (left) is addressed by linked solution
 The comment (left) is addressed by linked solution.

 The resolution strategy on top is combination of solutions on the bottom-right.

6.7.2 Compile and View Model

Resolution Strategy

P2_{S1} Use of the VE clock: The concept of *compile* or *build* is removed; the environment is always running, but time is stopped during an authoring action and can be resumed or restarted by the author at any time (or set to automatically resume after an authoring task). This allows the author to *pretend* to be the end-user to test the environment, while still being able to make authoring changes at any time.

P2_{S2} Combination of authoring and viewing interfaces: Authoring interface is accessible in addition to the end-user's interface. Objects are selected directly, or with different controls available on the console bar. Only authors have authoring right which determines whether or not authoring can be done.

P2_{S3} Changes are reflected immediately. Due to the WYSIWYG nature of the tool, changes are reflected straight away.

Interface Issue

P2 Compile & View Model

VE needs to be compiled or built before the end-result can be seen. This results in many problems.

P2 _{C2}	Lack of immediate Feedback
P2 _{C3}	Build All then View mentality
P2 _{C1}	Different representations
P2 _{C4}	Different controls for edit and view modes

Solution Set

VE clock (6.2)	P2 _{S1}
Unified WYSIWYG (6.2.2)	P2 _{S3}
Actions & Interactions (6.3)	
In-environment interface (6.4)	P2 _{S2}
3D Pointer (6.4.2)	
Object Select & Place (6.4.3)	
Revolving Menu (6.4.4)	
Console Bar (6.4.5)	
Speech Bubble (6.4.6)	
Obj-based Interaction Sp (6.5)	
Objects & Object Library (6.6)	
World & Avatar Objects (6.6.3)	

 The issue (left) is addressed by linked solution
 The comment (left) is addressed by linked solution.

 The resolution strategy on top is combination of solutions on the bottom-right.

6.7.3 Mixed 2D/3D Interfaces

Resolution Strategy

P3_{S1} Eliminating Application Switching: All desktop-based software is inherently 2D based and any desktop-based 3D virtual environment authoring tool is no different. A lot of mixture of 2D/3D interface switch is in the form of application switching between different tasks. Unifying all the tool into one would eliminate this mixture based on application switching.

P3_{S2} Shifting 2D interface to inside the VE: The rest of the 2D/3D mixture occurs with in the authoring tool itself, and this is resolve by shifting all 2D interfaces inside the virtual environment itself. This includes the pointer, menu and popup windows. The 3D pointer, revolving menu and speech bubble respectively address these three issues.

Interface Issue

P3 Mixed 2D/3D Interfaces	
Tools contain mixture of 2D/3D interfaces.	
P3 _{C4}	Reduced viewing area
P3 _{C6}	Build all then view mentality
P3 _{C2}	Constant switching between mindset
P3 _{C1}	Pointer confusion
P3 _{C3}	Cluttered interface
P3 _{C5}	Delayed update
P3 _{C7}	Difficulties in editing an object's attributes & behaviours

Solution Set

VE clock (6.2)
Unified WYSIWYG (6.2.3)
Actions & Interactions (6.3)
In-environment interface (6.4)
3D Pointer (6.4.2)
Object Select & Place (6.4.3)
Revolving Menu (6.4.4)
Console Bar (6.4.5)
Speech Bubble (6.4.6)
Obj-based Interaction Sp (6.5)
Objects & Object Library (6.6)
World & Avatar Objects (6.6.3)

P3_{S1}

P3_{S2}

 The issue (left) is addressed by linked solution
 The comment (left) is addressed by linked solution.

 The resolution strategy on top is combination of solutions on the bottom-right.

6.7.4 Multiple View-points Object Placement Interface

Resolution Strategy

P4_{S1} Two interchangeable viewpoints: Instead of the multiple viewpoints presented to the author in different panes, objects can be placed using the main 3D window already showing the VE. In addition to the usual first person viewpoints, author can *fly* above the VE to get a top-down bird's eye view of the entire VE. Authors have indicated that these two viewpoints should allow for sufficiently accurate placement of objects.

P4_{S2} Gravity and Collision Detector as independent monitors: other difficulties in the placing objects involving making sure objects do not cross into each other or the floor. These two functionalities are provided as independent monitors in the interaction subsystem.

P4_{S3} Object Selection and Placement Interface: Special objects placement pointers are presented inside the environment to assist author to accurately place objects.

P4_{S4} Automatic placement help based on object types: Objects can be set to have automatic reaction to being placed near to object of the same type (or other objects). For example, placing a wall next to the end of existing wall automatically connects them together.

Interface Issue

P4 Multiple Viewpoints Object Placement Interface

Placing objects involve authors monitoring different viewpoints

P4 _{C1}	Confusing interface
P4 _{C2}	Cluttering interface
P4 _{C3}	Difficulties in moving, placing & locating objects

Solution Set

VE clock (6.2)	
Unified WYSIWYG (6.2.2)	P4 _{S1}
Changeable Views (6.2.2)	P4 _{S1}
Actions & Interactions (6.3)	
Authoring Actions (6.3.2)	
VE-based Actions (6.3.3)	
Independent Monitors (6.3.4)	P4 _{S2}
Interaction Manager (6.3.5)	
In-environment interface (6.4)	
3D Pointer (6.4.2)	
Object Select & Place (6.4.3)	P4 _{S3}
Revolving Menu (6.4.4)	
Console Bar (6.4.5)	
Speech Bubble (6.4.6)	
Obj-based Interaction Sp. (6.5)	
Objects & Object Library (6.6)	P4 _{S4}
World & Avatar Objects (6.6.3)	

The issue (left) is addressed by linked solution
The comment (left) is addressed by linked solution.

The resolution strategy on top is combination of solutions on the bottom-right.

6.7.5 Selecting and Editing Objects

Resolution Strategy

P5_{S1} Direct Object Selection: Objects can be selected directly using the 3D pointer. Two interchangeable viewpoints also give authors an overview of the virtual environment. Alternatively the console bar also provides a link to a list of all objects in the scene which can be used to move the author to that object's location directly.

P5_{S2} Editing an Object with Speech Bubble: A speech bubble like interface is used to let authors change an object's attributes or edit its involvement in different interactions. The speech bubble can also help authors to keep track of the selected object.

Interface Issue

P5 Selecting and Editing Objects

Problems with indirect object selection.

P5 _{C1}	Confusing interface
P5 _{C2}	Difficult to use
P5 _{C3}	Keeping track of selected object

Solution Set

VE clock (6.2)	
Unified WYSIWYG (6.2.2)	P5 _{S1}
Changeable Viewpoint (6.2.2)	P5 _{S1}
Actions & Interactions (6.3)	
In-environment interface (6.4)	
3D Pointer (6.4.2)	P5 _{S1}
Object Select & Place (6.4.3)	P5 _{S1}
Revolving Menu (6.4.4)	
Console Bar (6.4.5)	P5 _{S1}
Speech Bubble (6.4.6)	P5 _{S2}
Obj-based Interaction Sp. (6.5)	
Objects & Object Library (6.6)	
World & Avatar Objects (6.6.3)	

 The issue (left) is addressed by linked solution
 The comment (left) is addressed by linked solution.

 The resolution strategy on top is combination of
 solutions on the bottom-right.

6.7.6 Lack of Real World Mappings

Resolution Strategy

P6_{S1} The World Object: The *World object* acts as an access point for authors to make environmental changes such as weather condition and time of days.

P6_{S2} Objects & Object Library: Objects in the library should act and react as close to their real counterpart as possible.

P6_{S3} Interaction handling: The *interaction manager* manages actions in such away that impossible actions are prevented. Examples of this functionality include:

- Actions are not repeated while test condition is true.
- Existing actions are interrupted if both the existing and new action can not be performed simultaneously.

P6_{S4} Gravity and Collision Detection: Both functionalities are provided via *independent monitors*. This allows author to use both to assist them in placing and manipulating objects.

Interface Issue

P6 Lack of Real World Mappings

Tools not reflecting real environment

P6 _{C1}	Lack of gravity & collision detection
P6 _{C2}	Minimal Object mapping
P6 _{C3}	Difficult or non-existent global environment specification

Solution Set

VE clock (6.2)	
Unified WYSIWYG (6.2.2)	
Actions & Interactions (6.3)	
Authoring Actions (6.3.2)	
VE-based Actions (6.3.3)	
Independent Monitors (6.3.4)	P6 _{S4}
Interaction Manager (6.3.5)	P6 _{S3}
In-environment interface (6.4)	
Obj-based Interaction Sp. (6.5)	
Objects & Object Library (6.6)	P6 _{S2}
World & Avatar Objects (6.6.3)	
The World Object (6.6.5)	P6 _{S1}

 The issue (left) is addressed by linked solution
 The comment (left) is addressed by linked solution.

 The resolution strategy on top is combination of solutions on the bottom-right.

6.7.7 Interaction Specification Interface

Resolution Strategy

P7_{S1} Object-based interaction specification: Interaction is specified by assigning objects as watcher and doer (which could be the same), objects are chosen to watch for a certain condition /event and then tell another chosen object, the doer to perform an action. This allows authors to focus on one interaction pair or link at a time. However, this introduces another problem of multiple of such pairs having its watching condition true and wanting a common doer to do impossible actions at the same time. This is handled by the interaction manager and is discussed in the next point.

P7_{S2} Interruption Handling: The interaction manager intervenes to prevent impossible actions from occurring. It follows the policy outlined in the previous section and matches quite closely to what is expected in real life. This affords the author to focus only on one interaction chain at a time.

Interface Issue

P7 Interaction Specification Interface

Problems with specifying interactions

P7 _{C2}	Avoidance of complex interactions
P7 _{C1}	Tools require extensive programming skill
P7 _{C3}	Reinforce multiple tools/modes mentality
P7 _{C4}	Reinforce compile all then view mentality
P7 _{C5}	Object Selection problem
P7 _{C6}	Locating & visualising interactively-linked objects

Solution Set

VE clock (6.2)	
Unified WYSIWYG (6.2.2)	
Actions & Interactions (6.3)	
Authoring Actions (6.3.2)	
VE-based Actions (6.3.3)	P7 _{S2}
Independent Monitors (6.3.4)	
Interaction Manager (6.3.5)	P7 _{S2}
In-environment interface (6.4)	
3D Pointer (6.4.2)	
Object Select & Place. (6.4.3)	P7 _{S1}
Revolving Menu (6.4.4)	
Console Bar (6.4.5)	
Speech Bubble (6.4.6)	P7 _{S1}
Obj-based Interaction Sp. (6.5)	P7 _{S1}
Objects & Object Library (6.6)	P7 _{S1}
World & Avatar Objects (6.6.3)	

 The issue (left) is addressed by linked solution
 The comment (left) is addressed by linked solution.

 The resolution strategy on top is combination of solutions on the bottom-right.

6.7.8 Time Manipulation

Resolution Strategy

P8_{S1} VE Clock & Clock-based Actions: The introduction of the VE clock allow author to manipulate the passage of time by increasing the clock speed. In order for this to work, all VE-based actions must happen with respect to this clock. To support rewinding, all VE-based actions started need to be logged by the interaction manager. Furthermore, independent monitors, whose operation is based on the VE clock if it is running or the real clock otherwise, allow for similar action to occur as authoring or VE-based actions. To assist authors further, whenever an authoring interface is invoked (e.g. Object is selected for authoring or when the revolving menu is invoked) the VE clock is stopped and is resumed when the authoring context is terminated.

Interface Issue

P8 Time Manipulation	
Lack of functionality to manipulate time during preview of VE.	
P8 _{C1}	Time wasting due to lengthy content
P8 _{C2}	Difficulty in previewing simultaneous actions
P8 _{C3}	Need for rewind, fast-forward & pause functionalities

Solution Set

VE clock (6.2)	P8 _{S1}
Unified WYSIWYG (6.2.2)	
Actions & Interactions (6.3)	P8 _{S1}
Authoring Actions (6.3.2)	
VE-based Actions (6.3.3)	
Independent Monitors (6.3.4)	
Interaction Manager (6.3.5)	
In-environment interface (6.4)	
3D Pointer (6.4.2)	
Obj Select & Place. (6.4.3)	
Revolving Menu (6.4.4)	P8 _{S1}
Console Bar (6.4.5)	
Speech Bubble (6.4.6)	P8 _{S1}
Obj-based Interaction Sp. (6.5)	
Objects & Object Library (6.6)	
World & Avatar Objects (6.6.3)	

 The issue (left) is addressed by linked solution
 The comment (left) is addressed by linked solution.

 The resolution strategy on top is combination of solutions on the bottom-right.

Evaluations & Verification

Appreciative Inquiry			
Discovery	Dream	Design	Delivery
Discovery Interviews	Consolidation Interviews	Design Cycle	Evaluation Interviews
			Verification Survey
			This Chapter

This chapter presents the results of the evaluation interview sessions and verification survey. Together they complete the design phase of the appreciative inquiry methodology. They also signify the starts of the delivery phase.

The evaluation interview sessions and the verification survey were to assess the level to which the solution and resolution strategies contribute to solving the interface issues outlined in Chapter 5.

An additional purpose of the survey was to present and evaluate the solutions and resolution strategies to the more general VE authoring population. This is necessary since a handful of authors from four VR applications took part in the discovery, dream and design phases.

7.1 Overview

This chapter presents the end-results of the Hands-on Evaluation process conducted with a group of VE authors. These authors were not involved in the conceptualisation and development processes of the solutions and resolution concepts. The chapter is structured into eight subsections, each evaluating an interface issue. In each subsection the following were discussed:

- Relevant comments concerning the interface issue that were made in the first phase²⁰ of the Hands-on Evaluation interviews.
- Relevant comments concerning the interface issue that were made in the second phase²¹ of the Hands-on Evaluation interviews.
- Results of the survey conducted where the interface issues and the resolution concepts were outlined.

For the presentation of the solutions and resolution concepts different versions of our interface demonstration tool called VE Builders were built. Several versions of VE Builder were used to demonstrate different aspects of the solution and resolution concepts. A manual (see Appendix A) containing an outline of solutions and how they work were also given to participants during the first phase of the Hands-on Evaluation interviews.

VE Builder

VE Builder is the name given to a set of applications that were designed to illustrate the various solutions to the interface issues outlined previously. VE Builder was programmed in C# and C++ using Microsoft Visual Studio.NET and the CsGL (C# GL) library [73]. For demonstration and consultation interviews, versions of VEBuilders were run on an IBM Thinkpad T30 with Pentium 4m 1.6GHz, 512Mb of RAM and an ATI Radeon 7500 graphics card. Three different versions of VE Builder were eventually coded to demonstrate different aspects of the solutions. All three versions have the revolving menu functionality but each has different entries.

- **VEBuilder I:**
 - Object Placement and Manipulation interface
 - Object Library
 - Weather and other environmental changes
 - Gravity and Collision detector (as independent monitors)

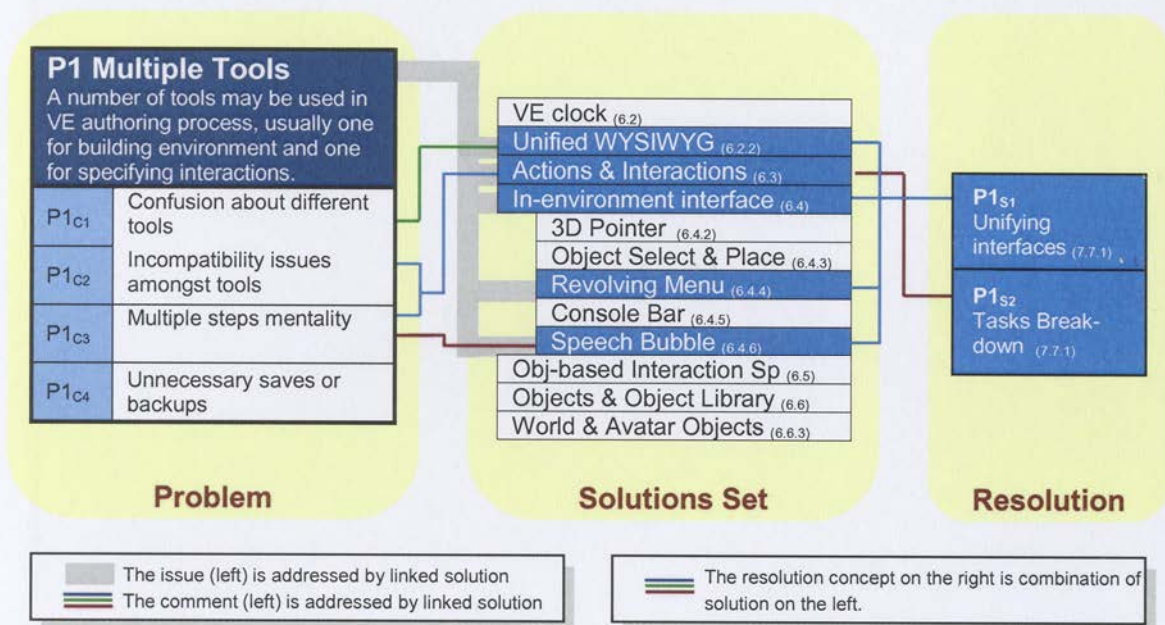
²⁰ The first phase involved the presentation and demonstration of the solution set.

²¹ The second phase involved the presentation and demonstration of the interface issues and resolution concepts.

- **VEBuilder II:**
 - Time Manipulation interface
- **VEBuilder III:**
 - Interaction subsystem
 - Interaction Specification Interface
 - Avatar/World Object

For the rest of the chapter, we will present the results of the evaluation process.

7.2 Multiple Tools



All authors indicated that combining environment building tasks and interaction specification tasks into one tool and one set of interfaces would be beneficial to the authoring process both in terms of learning and authoring. The process of learning would be simplified and streamlined as all tasks share a common interface.

Most authors saw the WYSIWYG interface as the most natural choice for an authoring interface. This combined with the in-environment interface to further cement the feeling of similarity across different tasks. Authors also reacted positively to the speech bubble interface and most liked the idea of personification of objects inside the environment with one commenting on the similarity to directing a play. Allowing authors to think of talking to objects while specifying interactions was one of the intended purposes of the speech bubble interface.

The concept of breaking down a task was a technical issue and as a result, most authors were not too concerned with it. They saw it as a behind the scene necessity

but most understood the concept after a brief explanation. However, they are not sure if it is a concept that authors would need to know.

In the survey, authors affected by the problem, all indicated that the solution should affect on them positively (see Figure 102 for the actual question and results). The comments were also generally positive.

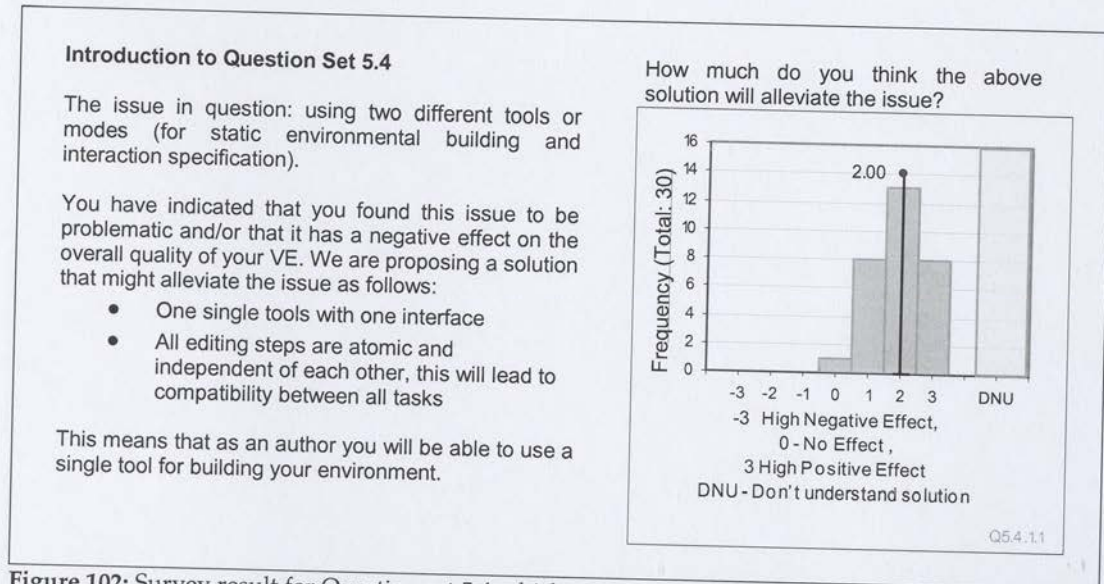
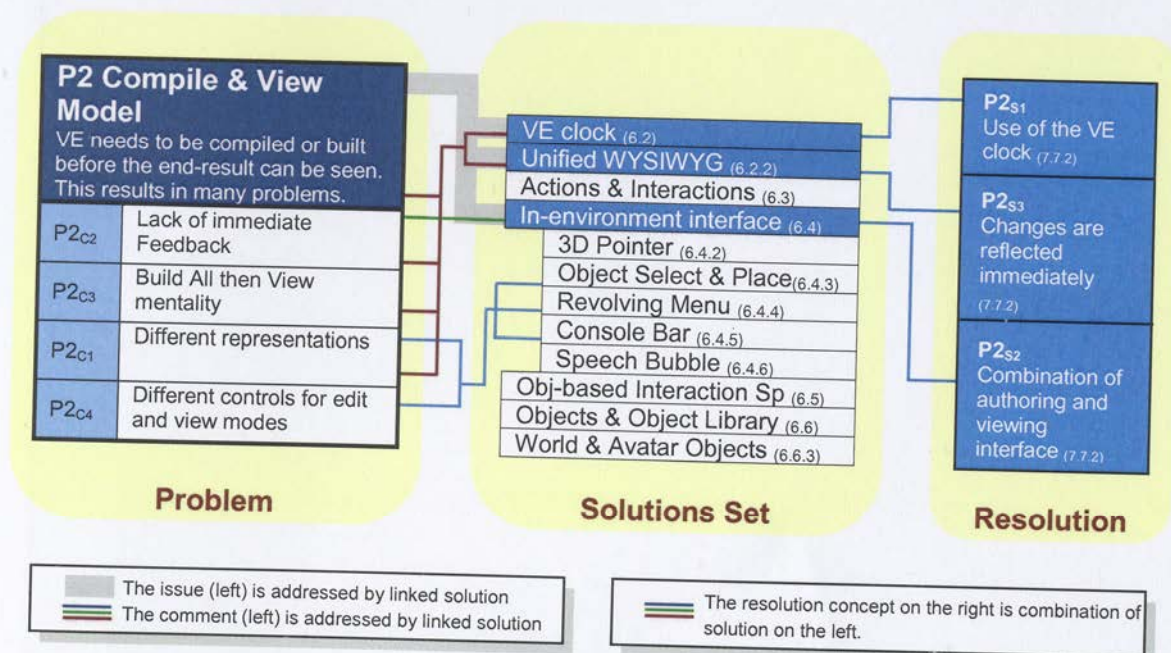


Figure 102: Survey result for Question set 5.4 which presented affected authors with a brief description of the resolution concepts for P1: Multiple Tools. Question 5.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.

7.3 Compile and View Model



Comments regarding the resolution concepts were generally positive. Although some authors thought that the VE clock could be confusing, many thought that it could be useful as it still allows authors, who were used to the compile & view based tools, to significantly reduce authoring time. This is because authors can quickly try different interaction routes instead of having to rebuild or restart after each change. They also liked the idea of combining authoring and viewing into one as well as the idea that they could just freeze time, make changes and resume at any time.

On the other hand, the resolution might extend the learning time as authors learn how to *author-on-the-go*, as one author put it, but this could have positive effect in that author effectively *preview* while editing. This means that potential problem might be picked up sooner.

In the survey, authors were generally enthusiastic about combining the authoring and viewing into one interface. Though some had experienced this type of interface before and mentioned a few problems including *perceived higher learning curve*, but most indicated that this was a better way. Figure 103 shows the actual text from the questionnaire and the relevant results.

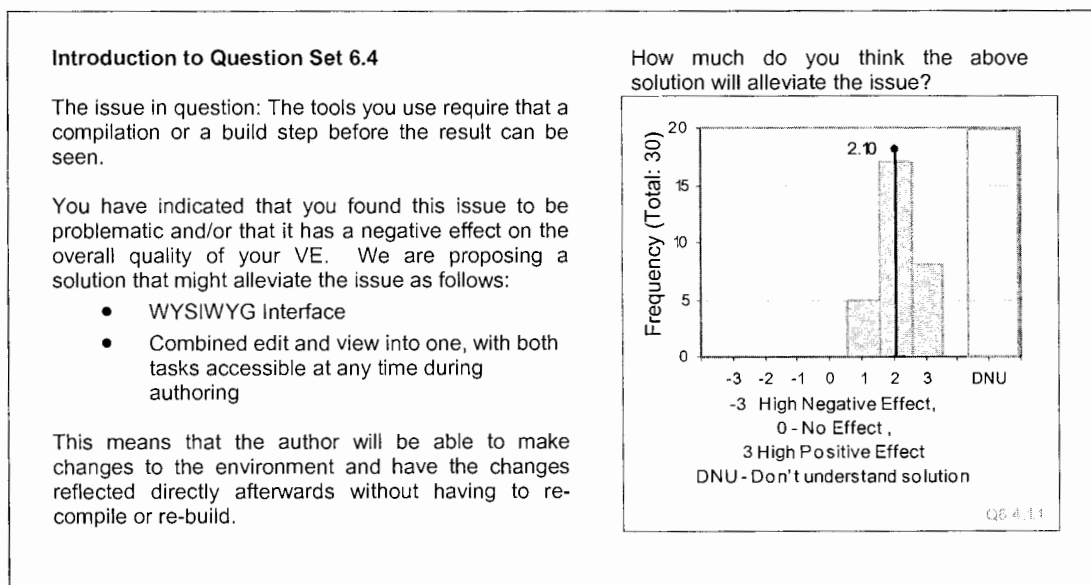
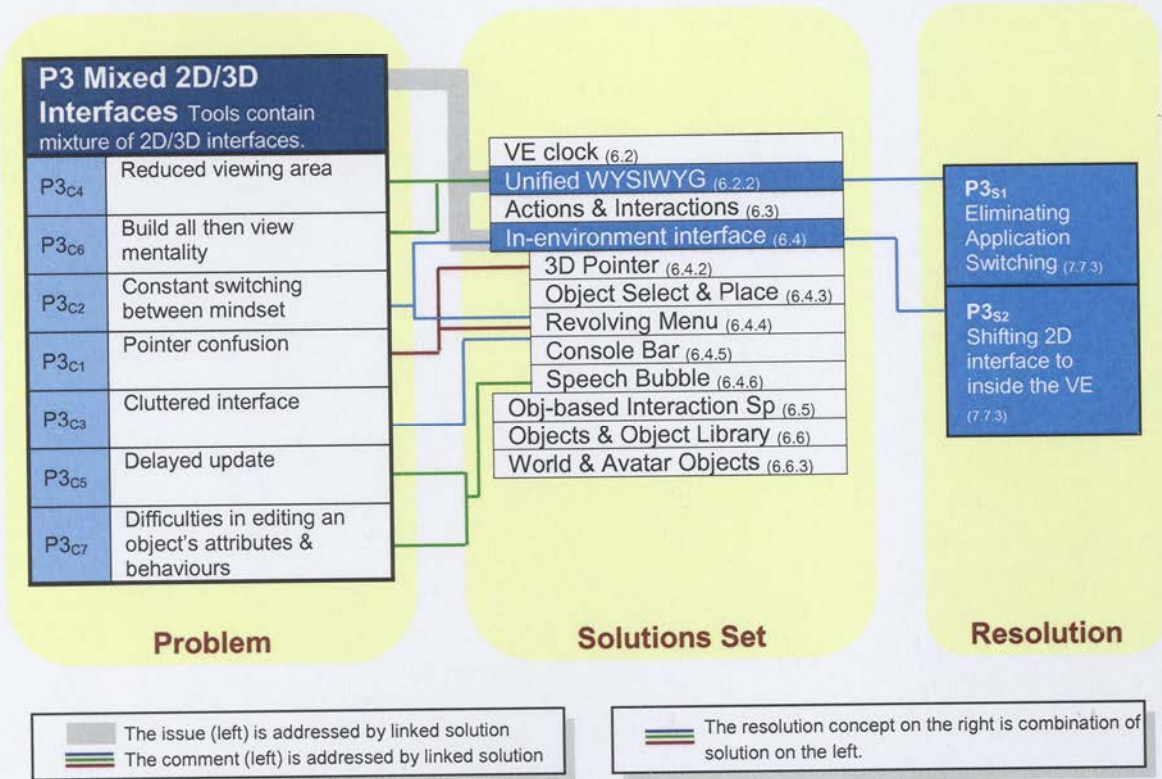


Figure 103 Survey result for Question set 6.4 which presented affected authors with a brief description of the resolution concepts for P1: Compile and View Model. Question 6.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.

7.4 Mixed 2D/3D Interfaces



Comments were also generally positive for this issue, although some authors did not see it as too problematic.

Authors liked the idea of the 3D pointer. Some were quite familiar with it and saw them as helpful in authoring virtual environments. Some even independently suggested representing the pointer with an avatar's pointing hand, something we suggested as a more advanced idea. Concerns were also expressed about the time involved in selecting a far away object as authors would either have to walk over to the position and use the limited range pointer to select it or go to the bird's viewpoint. Suggestions were made about pointing ability where authors can elect to use the pointer to point at a far away object to select it instead of touching it up close. With this improvement it was generally agreed that this method is better.

The revolving menu was also received well, but many found initially learning how to use it to be moderately difficult. Once mastered though, most agreed that it became easier to use with experience.

Most saw the speech bubble interface and were quite positive about the personification of objects. This, some say, would allow them to easily think about the objects and the interaction between them. More comments on the speech bubble interface can be found in 7.5 Multiple View-points Object Placement Interface.

Authors taking part in the survey also responded positively as shown in Figure 104.

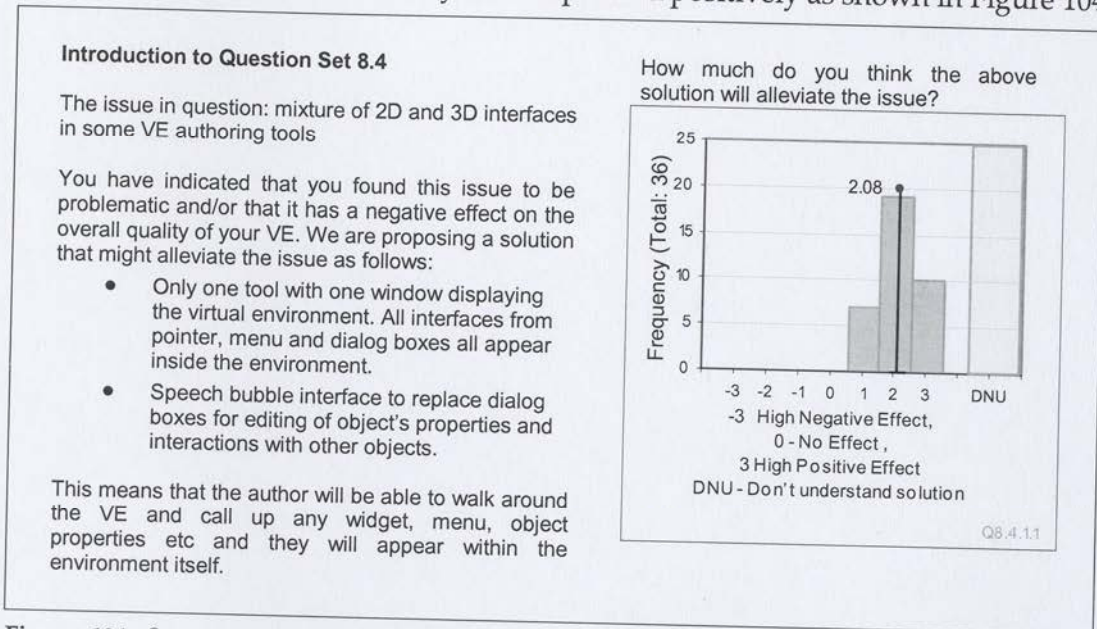
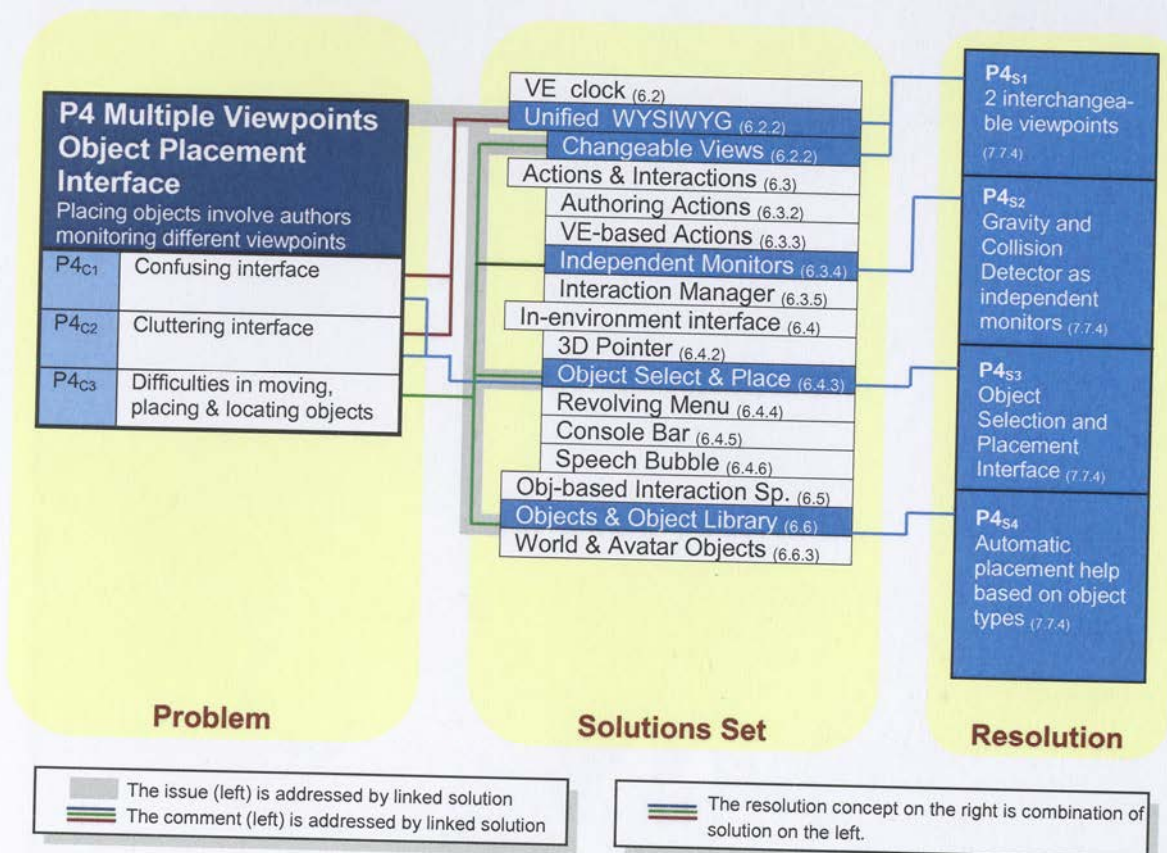


Figure 104: Survey result for Question set 8.4 which presented affected authors with a brief description of the resolution concepts for P3: Mixed 2D/3D Interface. Question 8.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.

7.5 Multiple View-points Object Placement Interface



All interviewed authors felt that our placement system was easier than the multiple viewpoints interface and that sufficient accuracy was provided for most cases. However, there were concerns about a minority of cases which demand more accurate placement of objects. They also highly rated the smooth change-over between the two-views, which was available during placement of objects. They said that it helped them to keep track of their (the avatar's) position and orientation when changing between views. The field of view in the bird's viewpoint was also found to be helpful.

Another set of features that authors liked was the availability of gravity and collision detection. This they said would be extremely helpful in placing object as well as time-saving as they would not have to fiddle to make sure objects did not go through floors. There were also concerns that sometime the features may not be needed, such as when objects need to float ²².

In first person mode, the restriction of only being able to place objects within arm's reach were felt as reasonable. However, advanced authors felt that it might be time-consuming if authors have to walk or run to the correct spot. Suggestions were made for allowing authors to run or allowing authors to pass through walls. Additional concerns were raised about automatic placements which, some said, needs to be thought out very carefully. Despite this most were fond of the auto-connecting walls. Some other examples were given such as automatic doors, windows or picture frames attaching to walls.

In the survey, authors were positive (see Figure 105) about the usefulness of our resolution strategy. There were many comments expressing the benefits of our interface.

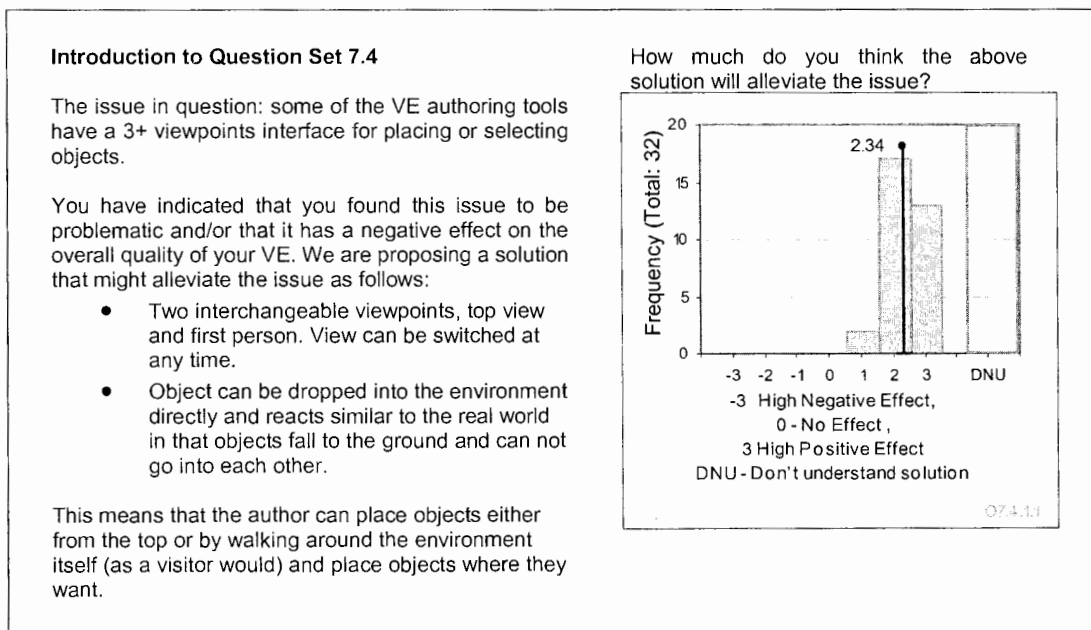
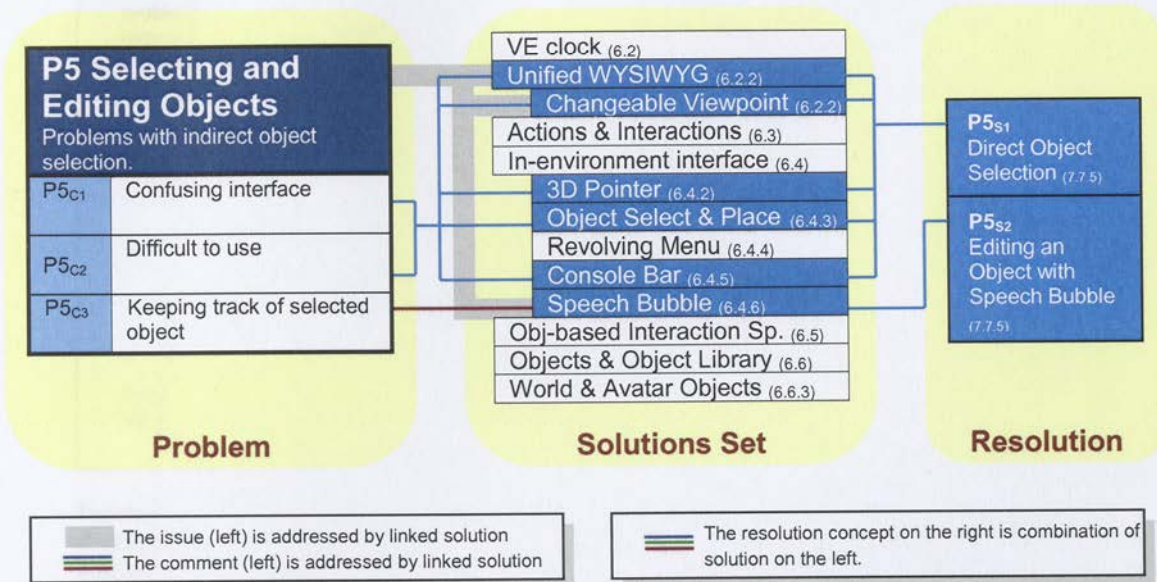


Figure 105: Survey result for Question set 7.4 which presented affected authors with a brief description of the resolution concepts P4: Multiple viewpoints object placement interface. Question 7.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.

²² Objects can be individually excluded from the effect of gravity and/or collision detection.

7.6 Selecting and Editing Objects



Similar points mentioned previously for the placement interface were echoed for selecting objects. This included the changeable viewpoints and the limitation of the arm's reach active selection/placement area.

The shortcut in the console bar was found to be quite useful for advanced authors, but it could lead back to the problem of authors having to remember objects names. Suggestions were made on grouping the objects according to distance from the avatar, geographical areas or the time they were added to the virtual environment.

The speech bubble interface was also well-received. Authors commented on the *user-friendliness* of the interface and how it was less intimidating than the usually grey dialog boxes. The benefit of the linkage between the object being edited and the object itself was mentioned. It was observed that many authors become engrossed in the editing and often forget which objects they are editing especially if there are many of them in the environment.

Another point often mentioned was the impression of personification of objects made possible with the combination of the speech bubble and other in-environment interfaces. One author mentioned that it reminded him of a role-playing game where players walk around and interact with different characters. Here, it was almost as though authors were walking around the environment and telling different objects how they should be and behave. He concluded that going with this route could make the VE authoring process more user-friendly because it allows authors to relate more easily to the virtual environment as they are interacting with it themselves. Relevant survey question and result are shown in Figure 106.

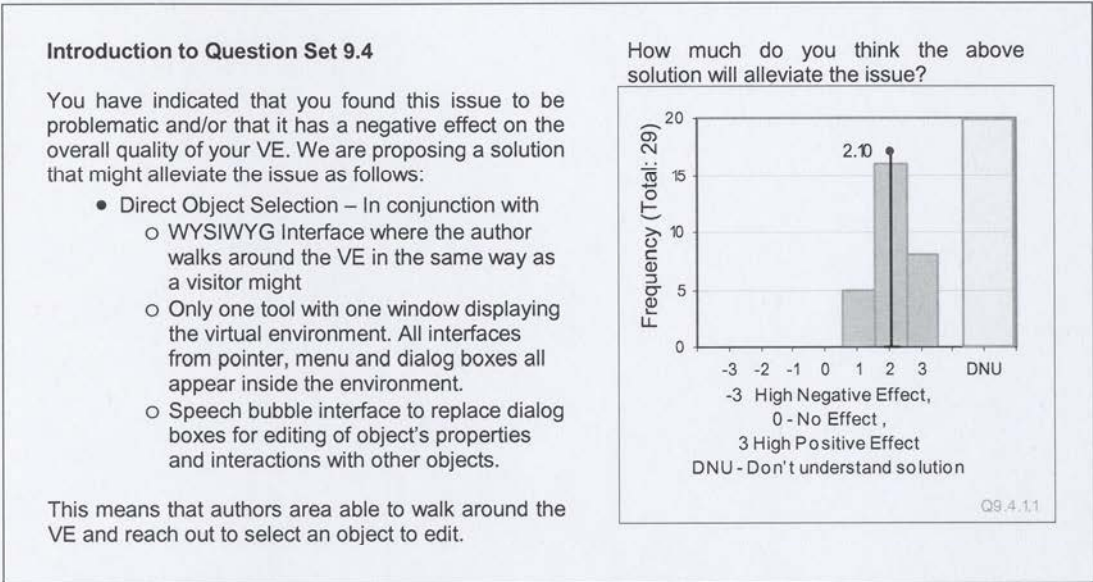
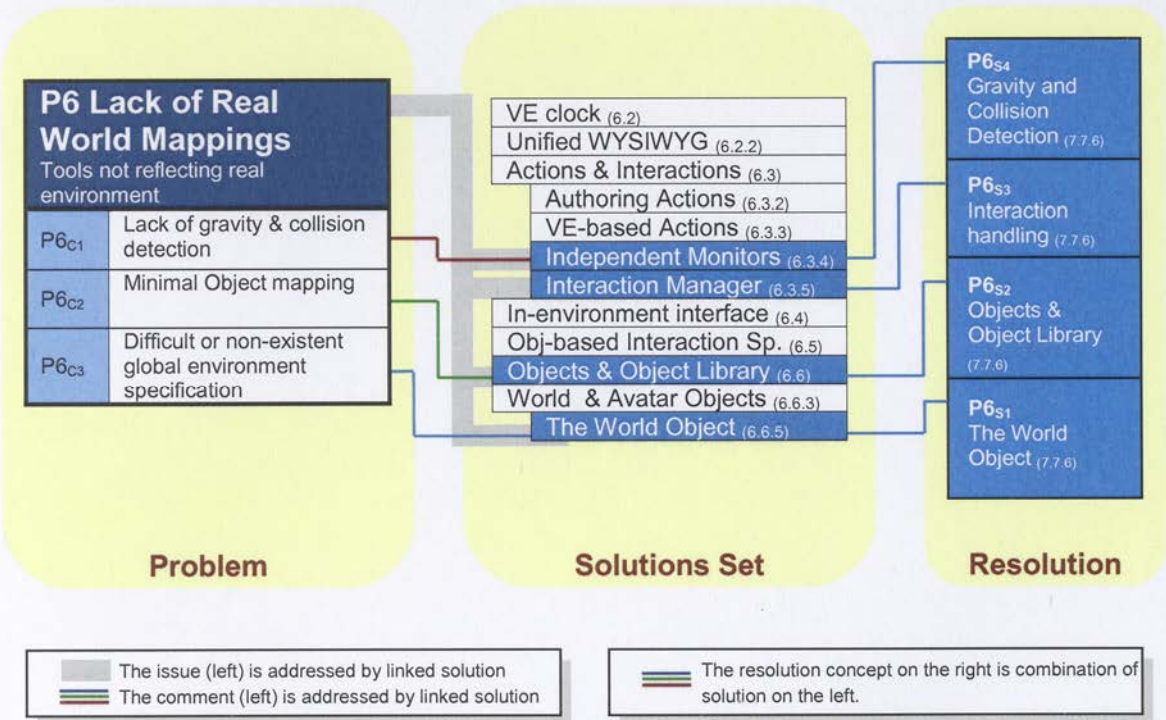


Figure 106: Survey result for Question set 9.4 which presented affected authors with a brief description of the resolution concepts P5: Selecting and Editing Objects. Question 9.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.

7.7 Lack of Real World Mappings



General comments concerning the overall resolution strategy were positive and authors were mainly enthusiastic about each individual solution. Most authors felt that if the environment were to behave the same way as the real world then they could use their intuition more during authoring. As one author put it, the word

intuitive depends a lot on how familiar one is with the tool. If the tool interface takes advantage of common knowledge, then it can only be advantageous to those using it.

On the issue of providing authors with a library of objects that appear and behave in a similar way to their real-world counterparts, authors expressed their dismay at why this was not the case with most tools already. Separating meshes/models and behaviours, as is the way with most authoring tools, seemed to be of no use to the authors. For example, when a virtual pet is dropped, all of its obvious behaviours such as walk should be readily available. In the survey, authors were also quite positive as show in Figure 107.

One author suggested an idea of freely available library of such objects, where any author can add objects or edit behaviours. This seemed to be a good way to build up such a library where objects can be downloaded and used freely.

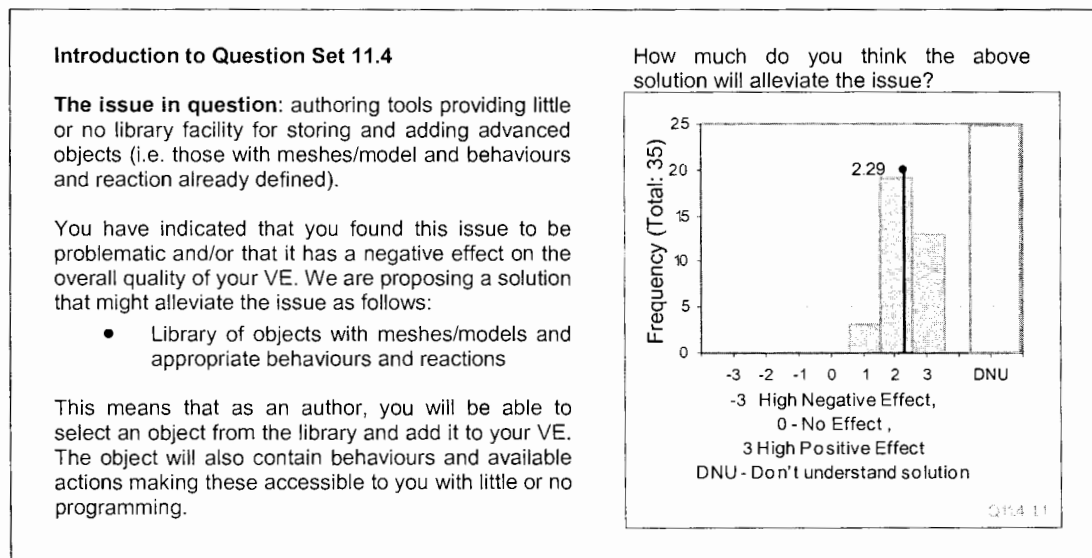


Figure 107: Survey result for Question set 11.4 which presented affected authors with a brief description of the resolution concepts P6: Lack of real world mappings (Objects' behaviours). Question 11.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.

Comments on gravity and collision detection were made concerning its positive effects in placing objects. Additional comments were mostly in relation to the ability to push objects around by other objects including the avatar (taking advantage of the collision detection system) or the falling behaviour via the presence of gravity. Others suggested improving the system further to include a more advanced physics processing system where objects can bounced off one another passing on their momentum. Independent monitors responsible for simulating gravity and collision detection could be used to provide further physics-based phenomena.

Authors also commented on being able to turn off the effect on a per-object basis, thus allowing the flexibility to have objects react in an *unnatural* way.

Survey participants rated this feature highly with most giving it the *high positive effect* rating as shown in Figure 108.

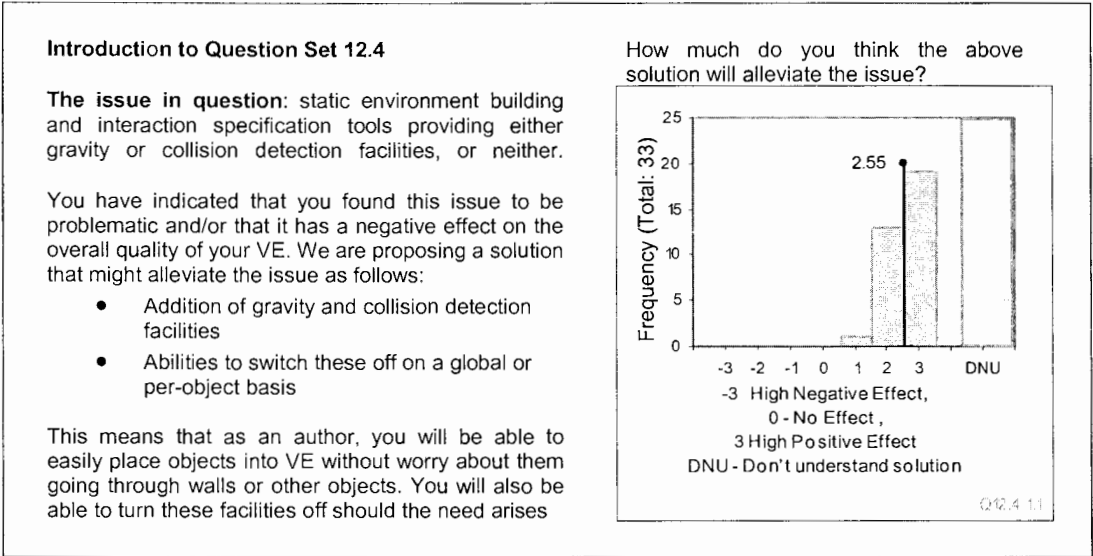


Figure 108: Survey result for Question set 12.4 which presented affected authors with a brief description of the resolution concepts P6: Lack of real world mappings (Gravity and collision detection). Question 12.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.

The third issue in the resolution strategy concerned providing authors with ways to manipulate their environment on a global basis, mirroring phenomena such as weather patterns, time of day and other surrounding environs. Authors were positive about this aspect but some questioned the ease of making one’s own weather condition or phenomenon. Most comments were on the time-saving factor, on the assumption that common weather and other conditions were readily available in the library. However, most authors interviewed were familiar with creating skyboxes or domes to reflect the outer landscape of their VE and would be satisfied if the specification could be done in a similar fashion. This usually involved specifying an image to be used as texture for the sky box. On the weather front, the usually foggy condition was indicated to be sufficient, but an author mentioned rain as another desired condition.

The related survey question also received positive responses which are shown in Figure 109. Comments were similar to those received in the interviews.

The solution aspect of interaction handling will be evaluated in the next section.

Introduction to Question Set 13.4

The issue in question: static environment building tool providing little or no support for easily creating or changing the backdrop, weather and/or other global properties.

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- Ability to change these global properties easily
- Availability of library of settings, time of day and weather.
- In conjunction with the WYSIWYG interface, you will be able to see the results immediately from the point of view of your visitor.

This means that as an author, you will be able to change the location, time of day or weather of VE easily.

How much do you think the above solution will alleviate the issue?

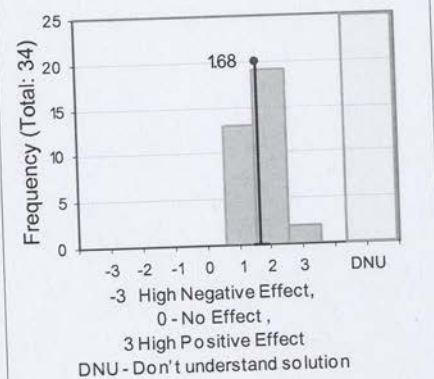
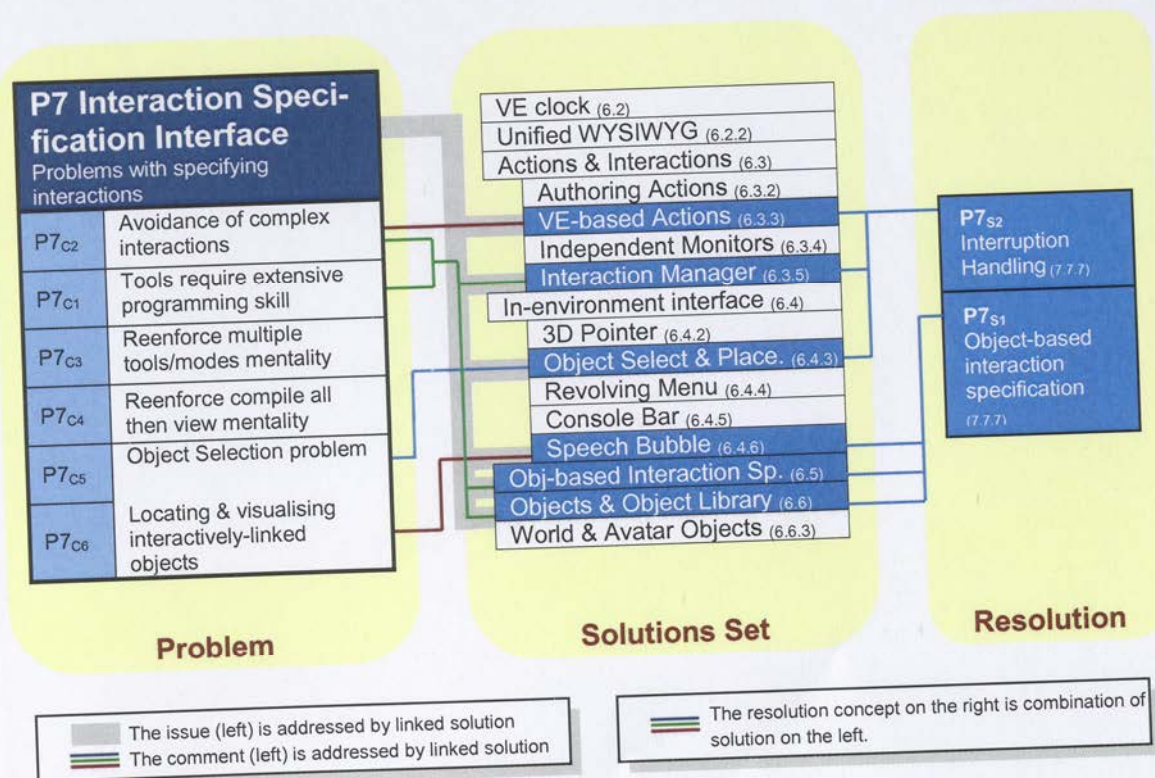


Figure 109: Survey result for Question set 13.4 which presented affected authors with a brief description of the resolution concepts P6: Lack of real world mappings (Global Environment). Question 13.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.

7.8 Interaction Specification Interface



Object-based interaction Specification

A few of the authors were familiar with object-orientated programming and straight away saw the similarity to that approach. Others were more enthusiastic about it, comparing it to directing a play, as authors could walk around the scene telling each actor how and when to act and react. One particular comment was on the effect of this issue and how it fit with other interfaces to complete the personalisation of objects in the virtual environments. She sarcastically argued that this would open up the tool to many authors since people are normally good at telling people what to do.

Negative points included the amount of time that would be needed to go through the process; as authors become more experienced, it would become more tedious to specify interaction that way. However, this could be something that we could develop further as authors use this system more and more. The best thing is perhaps to come up with a way that would bridge the gap between this interface and actual object-orientated programming/scripting languages that authors can learn as they go along.

Interruption Handling

The solution aspect concerning interruption handling received one of the most positive responses from authors interviewed. Main concerns from authors entailed striking the balance in controls that are in the hands of the author and that of the system. Most authors were happy with giving up the control that they have in interaction specification over to the system in exchange for simplicity as well as the focus on the current interaction the system afforded them. However, the system should not overstep in to silently guessing the intention of the authors in that it should not become '*too intelligent*' and derive any kind of priority order; that should be the author's job. At the very least, the system could point this out to the author. On this issue, concerns expressed were about correcting mistakes that could not be easily seen if the system were to do an interruption contrary to the author's original intentions. This would be difficult to see since the system would handle it automatically. A possible solution would be to indicate to the author that the system was handling the interruption and authors could then get a chance to see when this happens incorrectly or unexpectedly. One author suggested that at this point the system could ask the author to choose how the system should proceed and which interaction should take precedence.

In the survey, the complete solution with both object-based specification and interruption handling were presented to affected authors, all of whom gave it a positive score. Both the question and the result are shown in Figure 110. Comments made in the survey were generally positive.

Introduction to Question Set 14.4

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- Specification is done by selecting object directly, telling one to watch for certain events and react by telling others (or itself) to do something.
- Object Library with ready behaviours and reactions for common objects
- The system actively prevents conflicting or inappropriate action from happening (for example, you might two incompatible actions triggered off by different conditions being watched for).
For example you can tell an actor that:
 - When its 3pm, play the flute for 20 minutes
 - When someone's at the door, answer itWhen its 3pm, the actor will watch TV and any time within the 30 minutes, someone knocks on the door, the actor will stop playing the flute first, and not answer the door while still playing the flute.

This means that as an author, you will be able specify interaction by going to the appropriate objects and tells it to watch and react the way you want. The personification of objects should allow you to easily think about the interactions, while the system actively preventing impossible actions, allowing you to focus on specifying one interaction at a time.

How much do you think the above solution will alleviate the issue?

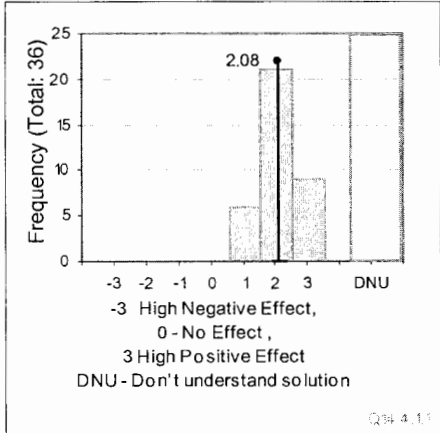


Figure 110: Survey result for Question set 14.4 which presented affected authors with a brief description of the resolution concepts P7 Interaction specification interface. Question 14.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.

Visualising Interaction Links

Functionality for visualising interaction chains was also well-received. Authors were positive about the ability to view an interaction chain visually but had some reservations about more complex interactions that could potentially look daunting or even cluttered. Survey results were also along the same lines (see Figure 111), with comments questioning the scalability of the system to more complex interactions.

Comments made concerning the speech bubble interface that displays the interaction chains in which the current object takes part in, were mixed. Positive comments were on the ability to view all chains at a glance. Negative ones concerned the inability to get more information from the interface and difficulties experienced in telling them apart. Suggestions were made on letting author's group related chains and that the system could group them based on different criteria such as creation time or on the other objects in the chain. In the survey, authors reacted along similar line with the result shown in Figure 111.

Introduction to Question Set 10.4

The issue in question: your authoring tool indicates interactional relationships between different objects via a programming language and that there are little or no visual links between them.

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- Visual Display of this relationship in top-view and first-person view

This means that as an author, you will be able to select and object and have their interaction link shown visually from the top where you'll be able to see the entire relationship.

How much do you think the above solution will alleviate the issue?

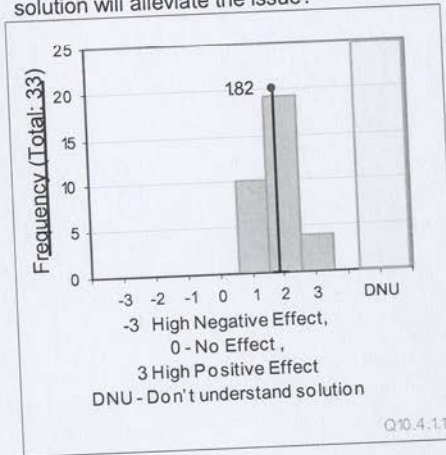
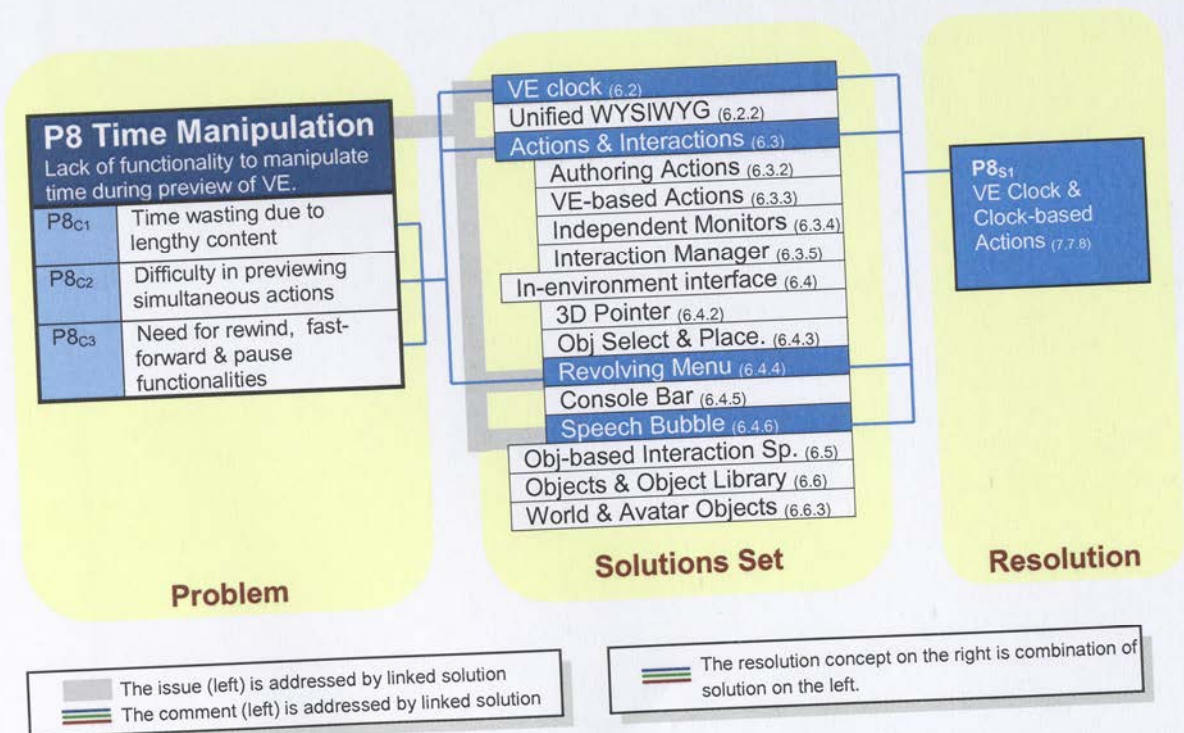


Figure 111: Survey result for Question set 10.4 which presented affected authors with a brief description of the resolution concepts P7 Interaction specification interface (Interaction link visualization). Question 10.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.

7.9 Time Manipulation



Time manipulation during authoring was on most authors' must-have list in the initial problem discovery interviews. The views of both the evaluation interviews and survey reflected this as most authors were optimistic about the effect this would have on their VE authoring, with time-saving being the most mentioned benefit. Others included the ability to try out different interactions quickly. Authors could start off with one then rewind and make changes and quickly see the effect. This could add a new dimension to the authoring process as authors would be more inclined to try things out. However, this could have a downside as authors may not have thought the interaction through properly before trying it out, and rely on just the perceived result on the screen which might not be telling the whole picture. The abilities to pause and slow time down were also given a thumbs-up, with authors readily seeing the benefits.

One of the negative comments concerned the confusion about undoing and rewinding. Some authors expected recently-added object to be removed when the environment is rewound. A few authors felt that authors will overcome this confusion with time and that the situation is acceptable given the benefit.

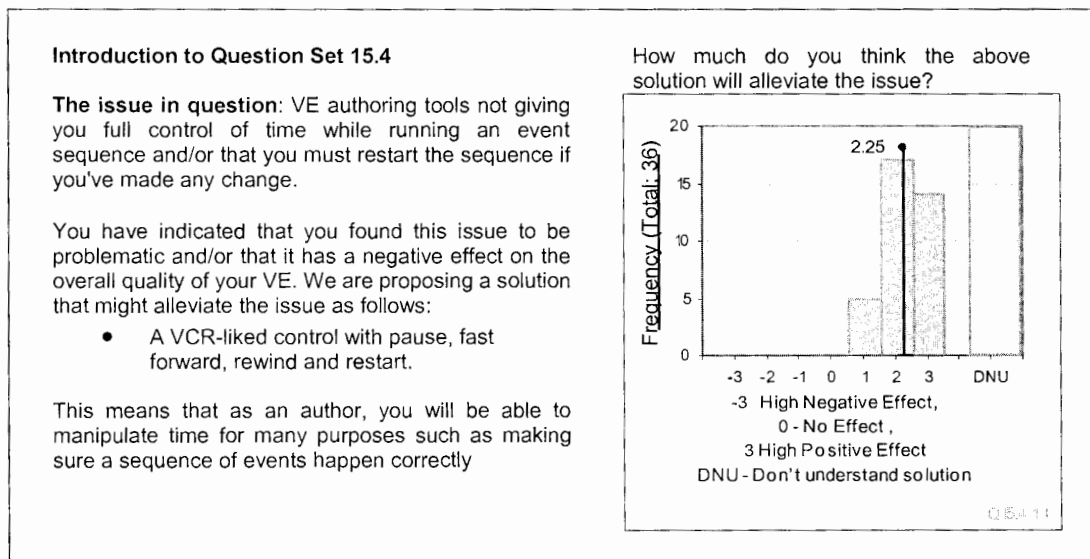


Figure 112: Survey result for Question set 15.4 which presented affected authors with a brief description of the resolution concepts P8 Time Manipulation. Question 15.4.1.1 asked the author to rate how they think this resolution would alleviate the problem.

7.10 Summary

In this chapter, we presented the results of the evaluation and the verification processes. The main purpose of the two processes was to independently assess the level that the solutions and resolution strategies resolve each of the eight interface issues. The evaluation process was based on hands-on face-to-face interview sessions with authors from the same four application areas as the original group which helped in the development, while the verification process involved an online survey which was opened to a wider group of authors.

The results of the hand-on evaluation interviews were generally positive, with authors indicating that they were satisfied that each solution set sufficiently addressed the corresponding issue. In terms of appreciative inquiry, these interviews played a part in establishing that the design model is in line with the dream model. There were also some negative comments which were recorded to be used in the next AI iteration.

The results of the verification survey are summarized in Table 2. It indicates the ratings for each resolution strategy, as well as the level at which the population were affected (i.e. those who indicated that they think that the issue in question could be improved upon).

Table 2: Summary of the results of the verification survey

Issues	Solution Ratings -3 High negative effect, 0 No effect, +3 High positive effect	Population Affected	p-values from 2 tail t-test comparing those who saw working solution* versus those who did not. (brackets gives no. of those who saw / those who did not)
P1 Multiple Tools	2.00	30/36 (83.3%)	0.5760 (7/23)
P2 Compile & View Model	2.10	30/36 (83.3%)	0.4669 (7/23)
P3 Mixed 2D/3D Interface	2.08	36/36 (100.0%)	0.2724 (8/28)
P4 Multiple View-points Object Placement Interface	2.34	32/36 (88.9%)	0.6755 (7/25)
P5 Selecting and Editing Objects	2.10	29/36 (80.6%)	0.4772 (6/23)
P6 Lack of Real World Mappings			
Object Library	2.29	35/36 (97.2%)	0.5840 (8/27)
Gravity & Collision Detection	2.55	33/36 (91.7%)	0.4626 (8/23)
Weather and other global properties	1.68	34/36 (94.4%)	0.6857 (7/27)
P7 Interaction Specification Interface			
Object-based specification	2.08	36/36 (100.0%)	0.3073 (8/28)
Visual Link Display	1.82	33/36 (91.7%)	0.7262 (7/26)
P8 Time Manipulation	2.55	36/36 (100.0%)	0.7810 (8/28)

A high proportion of the populations were affected by all eight issues and the solutions were well received and have all received positive ratings. Other issues that were not part of eight issues mentioned included sound and modelling related problems. We also found no significant difference in survey ratings given by those saw the working solution (those who participated in the hand-on evaluation process) and those who did not.

8

Conclusions

This chapter concludes this thesis and consists of three parts.

The first part presents a summary of issues and resolution strategies with respect to the virtual environment authoring interface. The second part deals with the lessons learnt from, as well as the suitability of, the appreciative inquiry methodology adaptation that was employed in our study. The last section of this chapter deals with how well our study has met the initial aims.



8.1 Virtual Environment Authoring Interface

8.1.1 Summary of Issues and Resolution Strategies

In this section we present a summary of the interface issues that were found to be in need of attention, as they significantly affected the authoring process, learning process and the overall quality of the resultant environment. We also present a brief outline of the solutions that were reached for them using the appreciative inquiry research methodology.

Multiple Tools: Some VE authors were forced to use different tools to perform different tasks. This was found to lead to confusion as authors frequently had to switch between them. In some cases, authors were forced to save their work between each switch, resulting in extra bookkeeping effort. Some incompatibility issues were also reported. The major consequence of this issue was found to be that authors using multiple tools thought that authoring VEs must be done in major steps, with each using one tool. We called this the *multiple steps mentality*.

The solutions that were reached for this issue involved WYSIWYG interfaces which were of the same look and feel for all tasks. On the technical side, we broke down all main tasks (represented by each class of tools) into smallest compatible tasks which allowed authors to do any of these tasks in any order.

Compile-and-View Model: Many tools forced authors to go through a *compile* or *build* step before the end result could be seen. Authors were frustrated because of the lack of immediate feedback from the system as to how their VE will look. This was often as a result of differences in representations of the VE (and in some cases, the control interface) before and after the build step. This also led to the *build-all-then-view* mentality whereby authors took a large building step to save compilation time.

Our solution involved allowing authoring to be done while the environment was running via the use of the authoring interface that exists within the environment. This led to a single interface for manipulating the avatar. Additionally, any changes were reflected immediately. The introduction of temporal manipulation functionality was also necessary since authors might need to *pause* the VE to make changes.

Mixed 2D/3D Interfaces: Most authoring tools encountered were based on two-dimensional display devices and as such most contained traditional two-dimensional user interface widgets such as toolbars, menus or cursors. Consequences reported by authors included reduction in 3D viewing area, pointer

loss and confusion, interface cluttering and difficulty and challenges relating to the two different types of interfaces.

Part of the solution to this issue was mentioned previously and involved shifting all 2D interface widgets to inside the VE itself e.g. we employed a revolving menu instead of a context menu and a speech bubble type interface instead of a dialog box. Unifying all the tools into one also eliminated application switching.

Multiple Viewpoint Object Placement: Most VE authoring tools employed an interface with multiple areas showing different viewpoints usually from the three major axes. Authors could use this to move and manipulate objects physically. This type of interface was found to be confusing and overly complex to many authors as well as cluttering up their work area. Authors also found it difficult to place objects this way as they have to monitor all three windows. Often they would end up with objects in the wrong place or inside another object.

Authors commented that only two viewpoints were really necessary – the overview of the VE from top-down and from the point of view of an end-user visiting the VE i.e. first-person view. Authors were further aided by the gravity and collision detection functionalities, preventing them from putting objects within other objects or the floor.

Selecting and Editing Objects: Authors reported difficulties in selecting objects, especially for interaction specification – some tools forced them to indirectly select object by name from a list or table. Some authors found this confusing to use, especially when there were many objects. Editing an object's attributes was often done in a window with no direct or visual link back to the object, leading to further confusion.

The proposed solution involved direct object selection using the two available viewpoints. In the first person viewpoints, authors could reach out and select an object, in a similar fashion to how an end-user might interact with that object. The speech bubble interface was employed when editing an object's properties, thus visually displaying the link between current object and its attribute display area.

Lack of real world mappings: This was a grouping of issues with concerns that authors' real world knowledge was not being explored by the authoring interface. The main issues were the lack of gravity and collision detection, the lack of an object library and the difficulties involved in creating peripheral effects in the VE such as weather. These functionalities, if available, would aid the authoring process tremendously as well as reducing the learning time.

In our solution, gravity and collision detection functionalities were provided and found to aid object placement as mentioned earlier. An object library providing objects with both physical appearance as well as typical behaviours was preferred

by the authors. Additionally, an interaction system that acts to prevent impossible actions was also incorporated.

Interaction Specification Interface: For most authors, interaction specifications were done via the use of programming or scripting language-based systems. This approach was reported to be overly complex and in some cases it led to authors avoiding creating complex interactions. Authors had to spend a long time mastering this approach. It was also found that some tools required a long and separate compilation step to incorporate the interaction specification into the VE.

We proposed an object-based specification interface, whereby authors can walk up to an object and literally tell it what to do. The speech bubble interface further enhanced this approach. We also incorporated an interruption handling facility which allows authors to not have to focus on stopping existing actions before applying a new one.

Time Manipulation: While testing a VE, authors often found it frustrating to have to repeat things in order to test possible branching or to wait for lengthy content to finish before proceeding to the next step. Another difficulty involved confirming that two near simultaneous events has occurred correctly.

A time manipulation functionality was introduced to allow authors to pause, fast forward or rewind their environment.

8.1.2 Problems Encountered and Potential Future Work

In this section we outlined issues that were not dealt with as well as other issues that arose out of our solution combinations. Within the framework of AI, these issues, if they are of high enough importance, would be looked at in the next iteration.

1. **Interruption handling:** In order to allow authors to deal with a number of simulated objects inside the environment, one can allow the author to give independent behaviour to each object, making things easier and more manageable. However, this means that there is a potential for conflicting instructions which the system has to resolve automatically during runtime. In our implementation we employed a last-come-last-serve resolution strategy whereby the newest instruction overrides all those still in effect. This is clearly not sufficiently flexible as authors might want to specify how conflicts should be dealt with.
2. **Object library:** According to many authors interviewed, the success of a virtual environment authoring tool depends on the number of objects already available in the library so that authors could just pick and drop any desirable objects into their VE. The object was then able to be

instructed to do actions without the authors having to reprogram it. This could save a lot of time and effort. However, one would need to know how such an object library could be built and maintained, how common actions could be specified and how one could perhaps build upon existing objects.

3. **Audio related interface:** An important contributing factor to an effective VE is spatial audio. Although, participants did not rate audio-related functionality as an important interface issue (in fact, it was mentioned only twice), it is increasingly becoming important in 3D environments. This can be clearly seen in the latest 3D game trend whereby convincing sounds play a role in setting up the atmosphere.
4. **Confusion between rewind and undo functionalities:** There was reported confusion concerning the rewind and undo functionalities, with some authors expecting undo to act as rewind and vice versa.
5. **Authoring rights:** The issue of authoring rights was raised by authors when introduced to the authoring within the environment approach. To some, the processes of authoring and the viewing of VE by the end-user were very similar, leading them to wonder if an end-user could accidentally be able to edit the VE. While we employed a simple flag to allow for authoring rights management, there may be a need to look at this issue closely in order to allow for more advanced functionalities such as collaborative authoring.
6. **Speech bubble interface:** The speech bubble interface that was employed as a solution to linking objects and an editing window might need to be looked at in environment where there are many objects that are close to one another. Perhaps in such cases, it might be clearer to have the tail of the bubble originating from the actual object. Occlusion of the bubble by closer objects might also be an issue.

8.2 Appreciative Inquiry Adaptation

8.2.1 Lessons from Methodological Adaptation

Appreciative inquiry has been used extensively in projects involving community and corporate environments, where the targeted group of people were somehow connected either geographically or within the same corporate structure. Despite this, we have successfully adapted this methodology to researching software-based user interfaces where users face the same software but are not geographically connected. Their only connection is the similarity in their jobs, in our case, virtual environment authoring. In adapting this research method to our needs, we learnt the following lessons:

Population: In the usual appreciative inquiry, the entire group of participants are available to take part in the exercise. In our case, this was not possible and as a result we performed the study on a sample group of virtual environment authors and then used a verification survey to infer the result for the population. We found this to be quite an effective way to apply appreciative inquiry to studies where the entire population of target personnel are not available to take part in the study.

Relaxed Interview atmosphere: The positive approach of the interview sessions led most participants to be quite relaxed. This had a positive effect as a friendly rapport was developed between the researcher and the participants. This not only helped significantly with the interview process, but also helped greatly with the later phases of designs and development.

Use of negative comments: Despite following the interview guidelines and the positive approaches, most comments were negative and might reflect that user interface design was not sufficiently user-orientated in the past. However, at the end of the evaluation phase, more positive comments were received indicating that the next cycle of appreciative inquiry would probably be more in line with other cases.

Negative comments were made use of as problems to be solved, or appreciatively speaking, as absent positive aspects to be strived for. This was fed into the design phase.

Dream Phase: Even after a few encouragements to 'come up with anything', it was felt that at first participants limited themselves during the dream phase by what was already available. They seemed tied down to existing user interfaces and the 'traditional' way of virtual environment authoring. However, most took up the challenge eventually and were able to provide many interesting ideas. The dream phase of the appreciative inquiry process became really useful as a launch pad to the design phase. In a way it is a free-for-all brain-storming period.

Development Cycles: A major departure from the original form of appreciative inquiry was the development cycle between the designing and delivery phases. This was where designs were prototyped and shown to the participants, at which point they provided feedback and any redesign suggestions were taken back and another prototype produced. One suspects that in the usual setting of appreciative inquiry, these cycles would not have been possible because of impracticality, for example, it would not be practical to try out different social approaches without affecting people's lives.

Split of evaluation from discovery phase: The split allowed each iteration to be treated independently as an evaluation is done at the end instead of at the beginning of the next iteration. This also made sure that an addition that results in

negative effects does not get lost. In the usual appreciative inquiry process, any negative-causing addition would have been 'ignored' in the next discovery phase, since any negative comments may not surface due to the positive interview style.

Positive Approach to Evaluation: Whilst the evaluation interviews were conducted in a positive manner, we could not help but wonder at the loss in objectivity of this part of the exercise. Although a different group of participants were used, surely not only the positive aspects should be focused on. In our evaluation method, negative comments were also welcomed.

8.2.2 Suitability of Appreciative Inquiry

We found that our adapted appreciative inquiry methodology worked well, with the main benefit being the creation of a positive and collaborating atmosphere amongst all involved. A few participants commented on the relaxed interview sessions. We believe that due to this aspect, many effective ideas and solution were developed.

It was felt that this methodology is more suitable when developing an entirely new interface to replace the old. e.g. where there are domain-experts readily available and are enthusiastic in coming up with better way of doing existing tasks. The success of the study depends on the willingness of participants. The positive approaches can often encourage this.

There is more work that needs to be done in order to evaluate the effectiveness of appreciative inquiry especially when compared with other methodologies.

8.2.3 Future Work for AI Adaptation

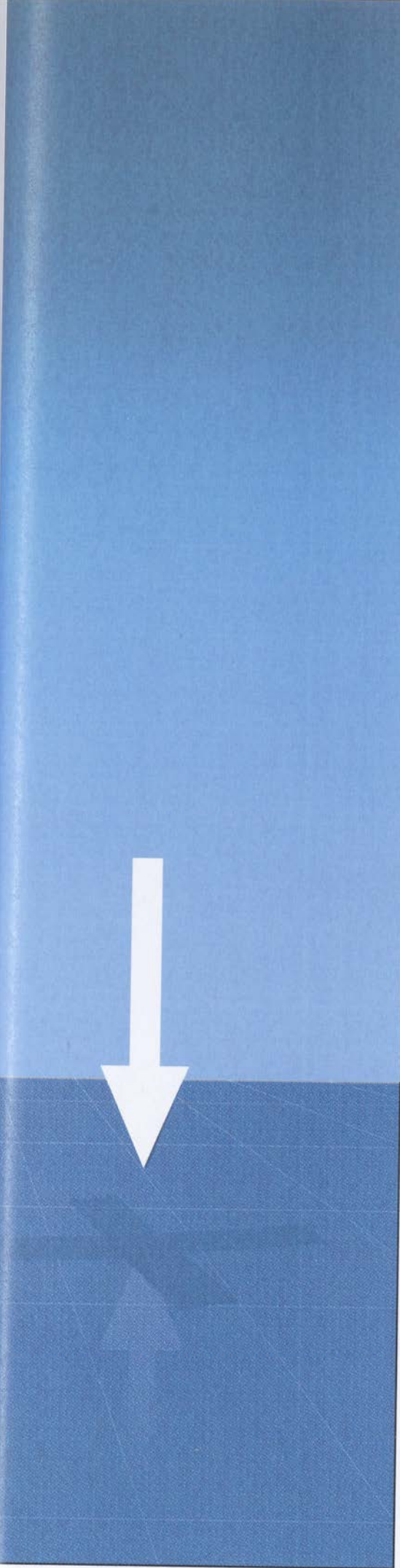
- a. **Further evaluation of Appreciative Inquiry:** This promising methodology could be employed in the area of user interface studies especially in areas where the task list has been extensively researched and reasonably established. Other areas where action research has been considered could also take this approach. With an increasing number of applications, the effectiveness (or lack of) of this method can then be clearly seen and properly evaluated.
- b. **Refinement of the development cycle:** The development cycle of design and evaluation phases can be improved upon. It might even be advantageous to run mini cycles of participatory action research type cycles during this phase.
- c. **Positive approach to evaluation:** As mentioned in the previous section, the positive approach to evaluation should be looked at in more detail. The

researcher needs to ensure that objectivity is maintained and that negative results of any changes are not lost.

8.3 Meeting the Aims

In relation to the project's main aims,

1. We were able to adapt the appreciative inquiry methodology successfully, using it to identify positive and negative interface issues experienced by the majority of non-VR-specialist virtual environment authors.
2. These experiences were integrated and together with further input from the authors, a set of prototypes and mock-ups of interface were built. This was then evaluated by a different group of authors. The second group of authors was satisfied that the changes improved the existing interface. Furthermore, the online survey which was open to any content-expert authors also resulted in similar findings. This indicated that the interface issues and solutions also applied to wider areas of VR authoring than just the applications area of those who took part in the earlier processes.
3. The adapted form of appreciative inquiry was employed and satisfactory results were achieved. However, further work is needed in order to further adapt this methodology to the field of computer science. This includes applying it to other areas where the methodology can be repeated in many iterations. The possibility of employing AI together with other methodologies, especially for evaluation purpose could also be explored.



Appendices

Appendix A: Using VE Builder Booklet

Appendix B: Survey Invitation E-mail

Appendix C: Online Verification Survey
Questions

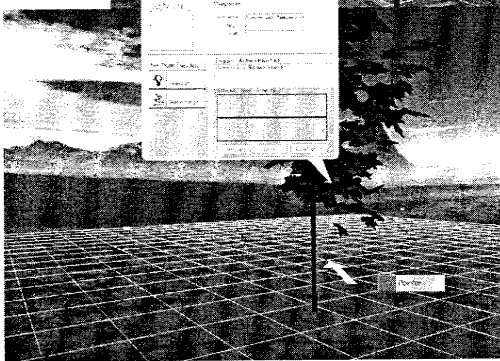
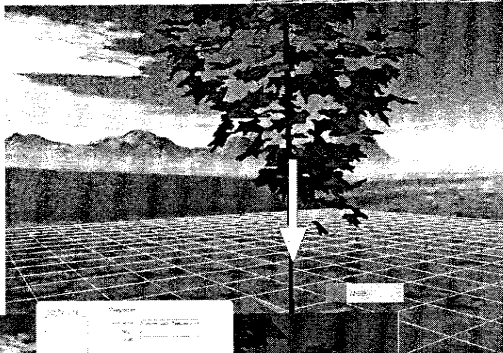
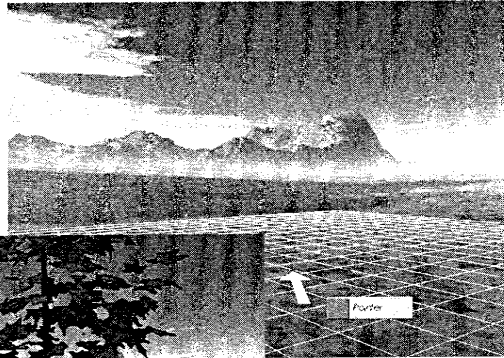
Appendix A: Using VEBUILDER Booklet

This booklet is used during the evaluation interviews in order to facilitate participant's understanding of how different features of the interface fit together. The document was originally in an A5-sized booklet form.



Using VE Builder

A Virtual Environment
Authoring Tool



Version 2.1a

Introduction

VR Builder is a tool aimed to provide start-up virtual environment (VE) authors who may not have extensive computer programming experience required by other tools. It provides a What-You-See-Is-What-You-Get (WYSIWYG) interface for authoring VEs. This means that you, as the author, are able to create a VE and also able to walk through it and see and interact with your creation just as your visitor would your finished product.

Authoring and Previewing your VE

As mentioned previously, as there are two main tasks associated with when creating a VE namely, authoring the VE and visiting or previewing the VE. With the WYSIWYG interface you are able to do both these tasks without leaving the tool. The system assumes your underlying VR system to provide at least two input buttons e.g. a mouse or a data glove.

While running the tool, you are able to interact with the VE just like your visitor would be. In fact, you are able to do whatever your visitors can in exactly the same way. This includes walking around the environment and interacting with objects etc.

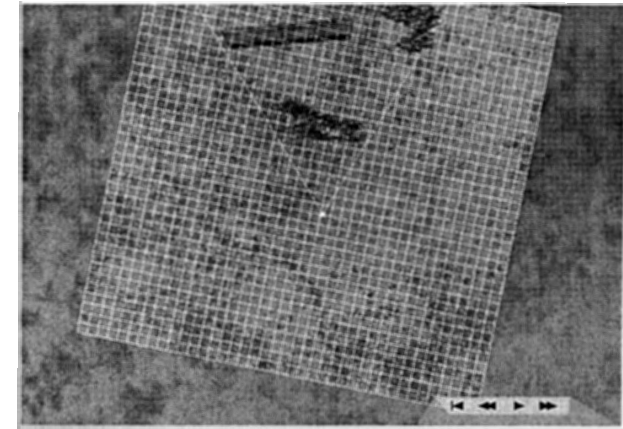
Available Viewpoints

There are two available viewpoints: the first person view and a bird's eye view. When editing, bear in mind that the first person view is the view that your visitor will see. In this viewpoint you will be able to do all that your visitor will be able to do and in the same way.



First Person Viewpoint ▲

Bird's eye viewpoint allows you to see the VE you are editing from the top down. The yellow dot on the screen in this viewpoint shows your location on the ground. You can also see the viewable portion and orientation which are indicated by the two yellow lines extending from your location point.



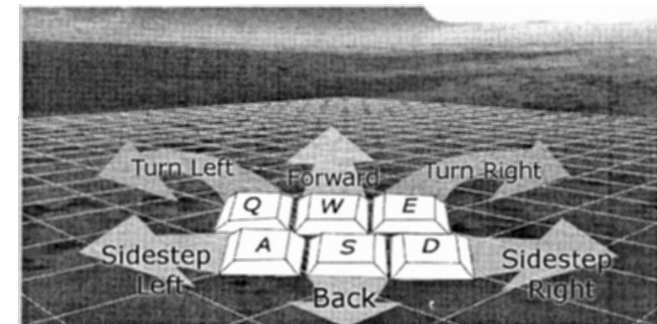
Bird's Eye Viewpoint ▲

The default view is the bird's eye viewpoint. To change your viewpoint, you may click on the 'change view' button on the bottom right hand-side or press 'R' on the keyboard.

Moving Around the VE

The VR Builder system when running on a desktop makes use of both the keyboard and a wheel mouse. The keyboard is used to move your 'body' and the mouse is used to move your 'arm' and 'hand' in the virtual environment.

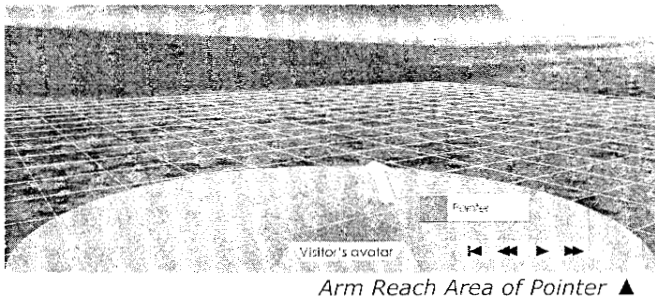
In both viewpoints you can your virtual **body** around using the following keys on your keyboard.



Keyboard Control for Moving the Body ▲

To move your virtual hand or pointer around, use the mouse. Move the pointer around like you would a 2D cursor. You may use your roiler to change the angle of the pointer, to point to lower or higher regions.

When in first person viewpoint, you are restricted to moving your pointer with in the *arm reach*. This is a semi-circular area around your *body*. If you need to reach far away object, move your *body* (via the keyboard) to make sure that the object is within your *arm reach*.



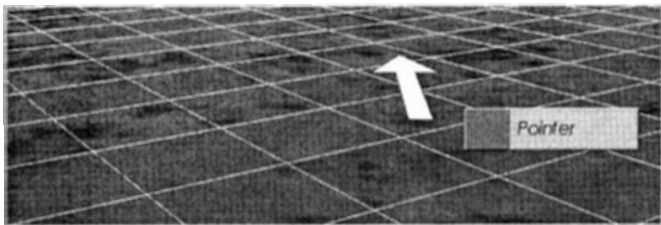
What Can Your Visitor Do?

It is important to always take into account what visitors to your virtual environment will be able to see and what they will be able to do. Your visitor will be able to:

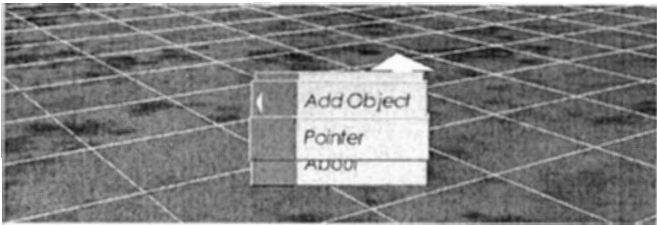
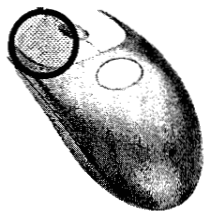
- See your VE in exactly the same way you do in the first person viewpoint.
- Interact with objects in your environment in several ways, including touching, touch and click (with both buttons) or even just coming near.

The Authoring Interface

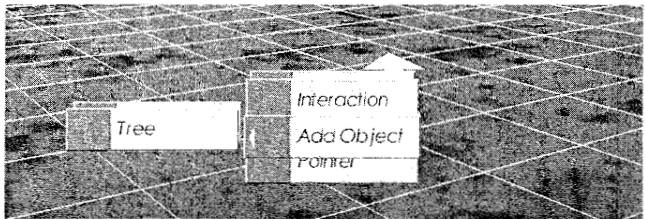
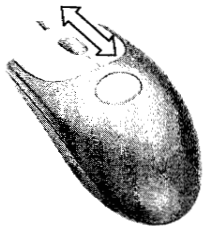
You may access the authoring interface as follows:

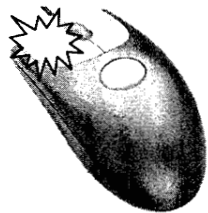
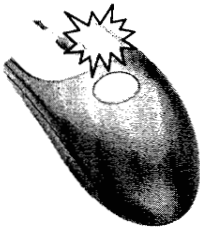
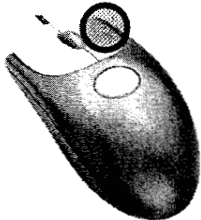


In order to access authoring interface, hold down the left mouse key until a 3D menu appear.



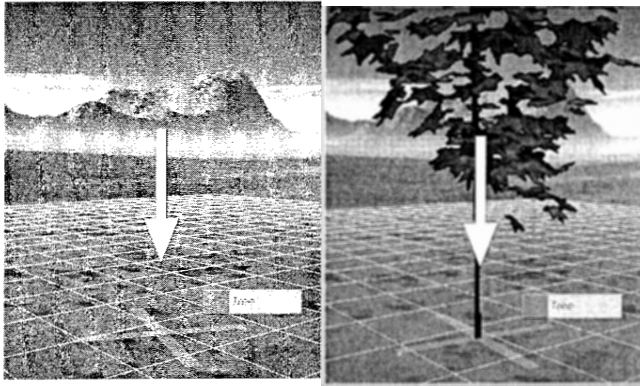
To move up and down the list, use the wheel on the mouse.



		
To select an item, click the left mouse button.	If you select an incorrect item and want to transverse back to the previous menu level, click the right mouse button.	To totally cancel the menu, hold down the right mouse key until the menu disappears.

Adding An Object

You can add new objects by accessing a top-level menu item "Add New Objects" then select the desired object. The pointer will then be transformed to the positional one. Move this pointer to the desired position and left-click to place the new object.



Note:

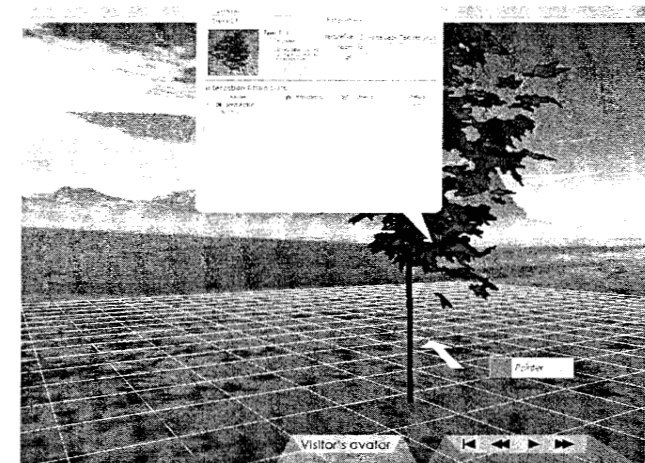
- When the positional pointer is shown, you will not be able to interact with other objects in the same way as you were with normal pointer.
- The object is automatically given a name. You may change this name by following the steps outline in the next section.
- Some objects may require you to indicate a starting point and an end point. In such cases, the first time you clicked will mark the beginning and the second for the end. If you want to cancel half way through, you may right-click on the mouse.
- Check the checkbox "Make others of my kind like me from now on"²³ and all objects of this type to be created afterward will have the current appearance of this object. To cancel, uncheck the box on any of the object of the same type.

²³ See FAQ for full detail on this

Modifying Objects' Appearances

After an object has been added to the world, you may modify its appearance such as type, height or colour. This can be done by following the following steps:

1. Place your pointer near or in your target object; it will then be highlighted.
2. At this point, hold down the right mouse button until a speech bubble appears from your selected object.



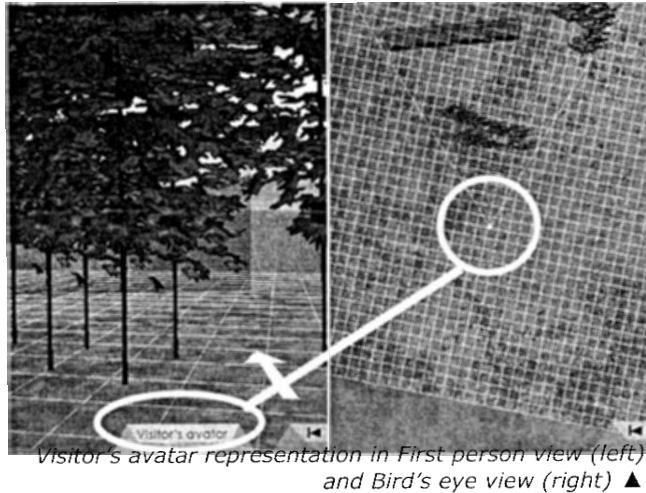
Speech bubble of the select object ▲

3. You may then make any desired adjustment, noting that the object changes as you make your adjustment. During this time, you may walk around the VE or change view.
4. When you are done, hold down the left mouse button, until the speech bubble disappears to permanently adjust the object's appearance or hold down the right key to return to the object's previous appearance without updating.

Note:

- You may need to alter the angle of your pointer to select different objects including the floor.
- If you hold down the mouse right button without entering an object, you can change the VE global properties such as the surrounding, the weather and other factors.

- A yellow block on both viewpoints²⁴ represents your visitor's avatar²⁵. You may change the avatar properties (such as height, walking speed etc) by calling its speech bubble.



The Speech Bubble Interface

The speech bubble interface, mentioned in the previous section, is used extensively to modify an object's properties and interactions among a group of objects.

There are three sections as follows:-

Section 1: Common Object Properties

This is located on the top left corner of the bubble. It contains properties that are common to most objects. These are:-

Name: The name the object is known by

Type: The object type.

Object Name
Tree001

Type: Tree
☐ Invisible
☐ Make other object of my kind like me from now on

Delete Object

Properties

textureFile D:\Home\Jack\Textures\youn

height 4

width 1

Interaction Chain List

Name	Watchers	Doers	Status
1 Billboard Action	1*	1*	Enabled
Add new ...			

This is not changeable by the user

Invisible: Whether or not the object is visible to the visitor. An invisible object may still interact with other objects. Object invisible by default include sound objects.

Make other object of my kind like me from now on: Checked this box so new objects of this type to be created will be like this objects both in terms of appearance and self interactions. More detail can be found in the FAQ.

Delete Object: You can delete the object from the VE by clicking here. However, if this object is part of an interaction chain, you will need to specify what happens to the rest of the chain. See the FAQ for detail

Section 2: Type Specific Properties

This is located on the top right of the speech bubble. It contains type specific properties of the object such as height and colour.

Object Name
Tree001

Type: Tree
☐ Invisible
☐ Make other object of my kind like me from now on

Delete Object

Properties

textureFile D:\Home\Jack\Textures\youn

height 4

width 1

Interaction Chain List

Name	Watchers	Doers	Status
1 Billboard Action	1*	1*	Enabled
Add new ...			

Section 3: Interactions

The last section is located on the bottom of the bubble and is concerned with interactions that this object is a part of. The section consists of the *Interaction Chain List* which contains all the chain in which the selected object is a part of.

An interaction chain is used to specify an

Object Name
Tree001

Type: Tree
☐ Invisible
☐ Make other object of my kind like me from now on

Delete Object

Properties

textureFile D:\Home\Jack\Textures\youn

height 4

width 1

Interaction Chain List

Name	Watchers	Doers	Status
1 Billboard Action	1*	1*	Enabled
Add new ...			

²⁴ In the first person view it is right near the bottom edge of your screen.

²⁵ Visitor's representation in the VE.

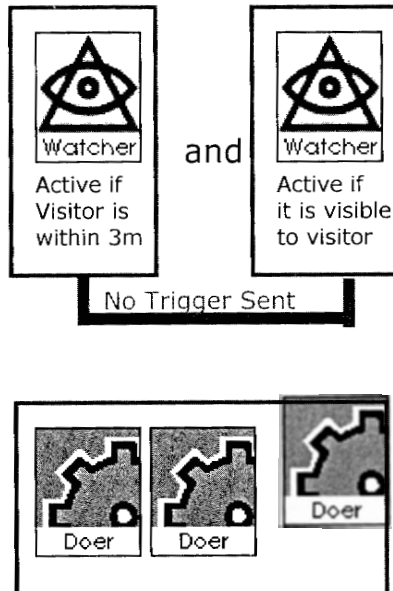
interaction. For example, you may want to create an interaction whereby if a visitor clicked on a switch you would like a distant light bulb to turn on; in this case, the switch is known as the **watcher** because it *watches* for certain event or action from the user and then trigger another object to *do* an *action*. The object that ended up performed the desired action is called the **doer**. Together the trigger and action objects form an **interaction chain**.

The rest of the interface concerning interaction chain will be explained in the next section.

Adding Interaction

Before you can add an interaction, it is necessary to present you the concept of watcher-doer interaction chain in more detail.

In general, a watcher watches for certain event. When this event occurs, the watcher is said to be in its **active** state. Similarly when the event it is watching for is not happening, it is said to be **inactive**. However, they may be a group of watcher involved in some logical way; for example, you might want to say that if the user is within 3m and if it is visible to the user. Depending on the active status of the watchers involved, a trigger may be sent to the doers. The doer is said to be in an active state if it has been triggered to do its assigned action. The diagram below illustrates this idea.

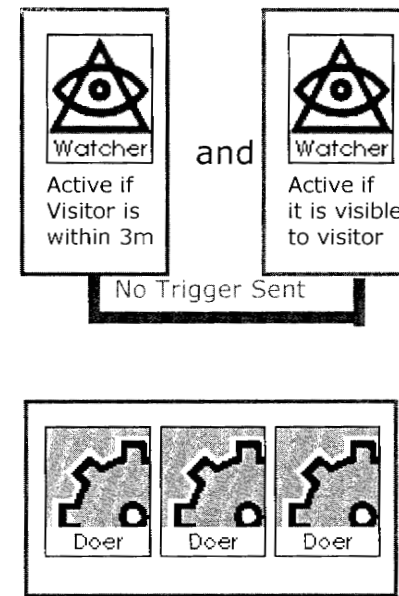


VE Situation:

Visitor is not within 3m of object, nor is it visible.

Interaction Chain Situation:

All doers are inactive since the watchers are all inactive.

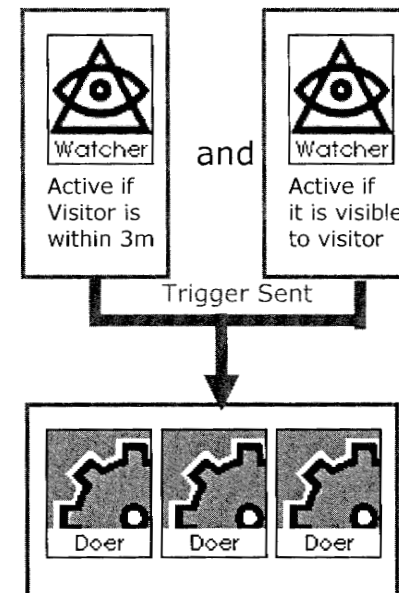


VE Situation:

Visitor is not within 3m of object, but it is visible to the visitor.

Interaction Chain Situation:

Only one watcher is active. A trigger is only sent if watcher1 **and** watcher2 are active. Hence all doers are inactive.



VE Situation:

Visitor is within 3m of object, and is visible to the visitor

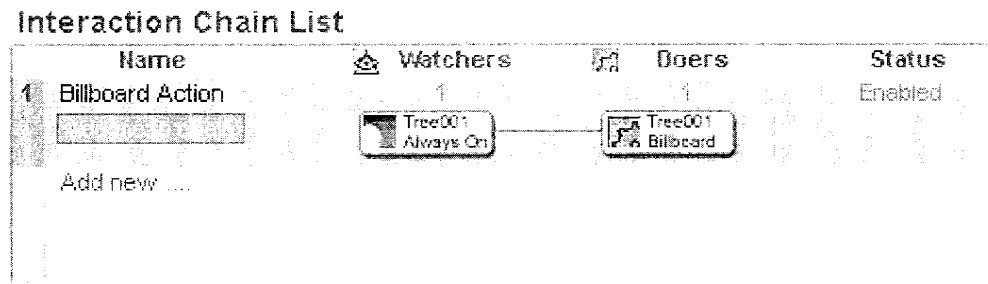
Interaction Chain Situation:

Since both watchers are active, a trigger is sent to the doers. Hence all doers are now active.

Interaction Chain List

An interaction chain may involve more than one of each type of objects or only one object where that single object acts as both the watcher and the doer.

The interaction chain list typically looks like this one below. You may select an item for inspection by clicking on it; the selected item expands to give more information.



The follow fields are shown on the list.

Name: This is the name that you gave on creation of the chain (or a default name if you did not give it one).

Watchers: For an unselected item in the list, this shows only the number of watchers in the chain. The number is red if none of the watchers are active, and green if at least one is active. A star next to this number indicates that the current object is participating as a watcher in this chain. On the expanded item, all watchers and doers are shown in more detail as follows.

Icon indicating the type of watcher/doer

Name of object that is a watcher/doer. Star next to a bold name indicates that the object is the currently selected object.

Background Colour:
Red: Inactive: watcher has not triggered or doer is not doing the action.
Green: Action: watcher has been triggered or doer is doing the action

Doers: For an unselected item, this shows only the number of doers in the chain. This number is red if the doers are not active and green if they are. A star next to this number indicates that the current object is participating as a doer in this chain. An expanded chain also shows doers in the similar way to watcher as described previously.

Status: This indicates the current status of the chain. A chain's status may be disabled, enabled incomplete or edit. A chain is incomplete if does not have any doers. However a chain is complete if it has no watcher; in such cases the doer will inherently be the current object and it will constantly do the assigned action. The status of edit is displayed when the chain is being edited.

Types of Watchers

There are standard types of situation that a watcher maybe set to watch out for. Object usually watches out for other objects (called target object which could include the visitor's avatar) and/or their actions. The different available types of watchers in this version are:-

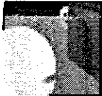
Proximity watcher: The watcher watches for when the target object is in certain range.

Target Object: Any
Option: Active if target object is *{Closer than/further than}*²⁶ *{distance}* metres.

Collision (touch) watcher: The watcher watches for when the target object collides or touches it.

Target Object: Any
Option: None

²⁶ {choice1/choice2} This means you will be given a drop-down list of these choices. A plain {name} means you'll be able to assign value for these variables.



Visibility watcher: The watcher watches for when the target object is within its visibility range i.e. when it can see the target object. For this version of VE Builder, only the visitor can be assigned as a target object.

Target Object: Visitor
Option: None



Property change watcher: The object watches out for any changes in its own properties i.e. the target object is the target itself. These are properties that are already changeable by the author via the speech bubble.

Target Object: The selected object itself
Option: Any updatable properties of current object.



Primary and secondary click watchers: The object watches out for when the visitor click on it. Primary click is usually left-clicking with a mouse and secondary is right-clicking.

Target Object: Visitor
Option: None



Time Watcher: The object watches out for certain time (global time) for example, 3 minutes after the start of the world.

Target Object: None
Option: Active {time} seconds after {now/ since world start}



Always On Watcher: This watcher always sends out an active signal. It should be used when you want a doer to do certain action all the time. The chain is only allowed to have one of this and not other watcher.

Target Object: None
Option: None

Types of Doers

An object may do any action which was given to it by object or library authors for example, a cat object may be able to eat, make sound, sleep, walk or run. These are type-specific actions and are different for each different object types. However, there are non-type-specific actions which include:

Making changes to properties: These are properties which are already changeable by author via the speech bubble interface, for example, changing the colour of the object or making it invisible.

Target Object: The selected object itself

Option: Any updatable properties of current object.

Change position and/or orientation: You may set the object to move to or face a particular orientation.

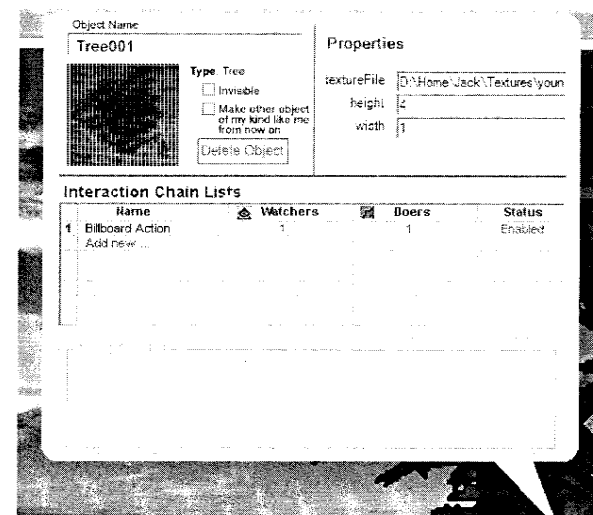
Target Object: Any

Option: {move towards target object / face target object / orient to the same as orientation as target object / move to {position} }

Creating an interaction chain

You will need to identify your watcher and doer objects. There is no order in which you have to follow, you may define either type of object first, follow by another, and then still go back to the first object to make further changes.

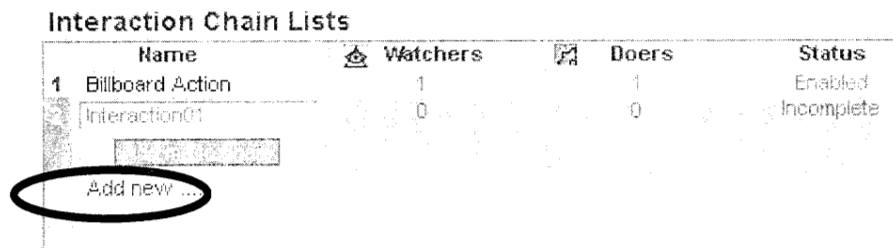
1. Enter an object and then hold down the right mouse button until a speech bubble appears (as you



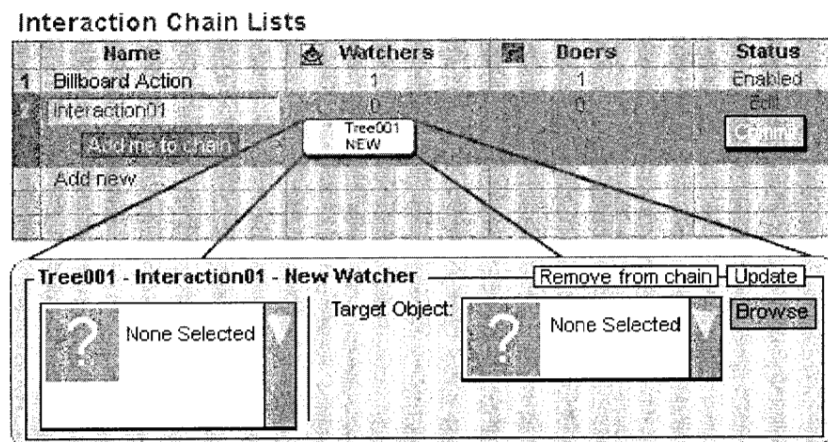
would do when you want to modify its appearance).

In the speech bubble you'll see a clickable 'Add new' in the 'Interaction Chain Lists'; click on it. You may give this chain a name for future reference or you can accept the default name.

- After creating a new chain, you will see a button underneath the name labelled 'Add me to chain'. On selection a menu appears with choices 'as a watcher' and 'as a doer'. Select the desired choice.

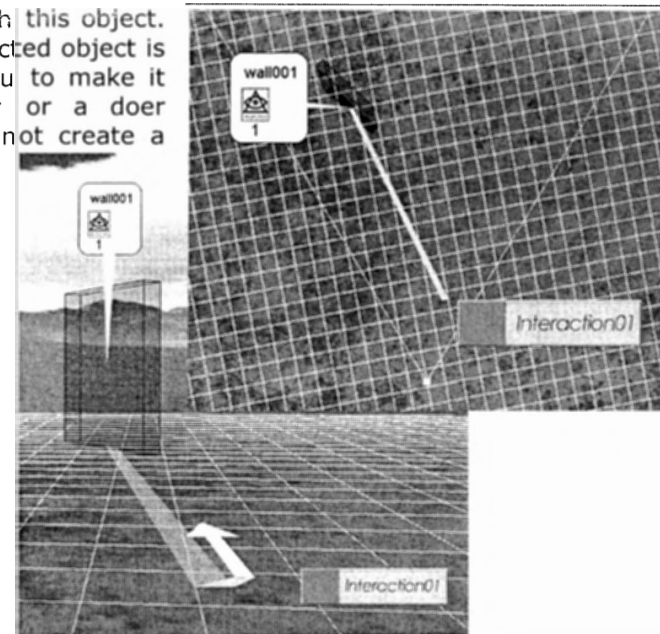


- The bottom part of the speech bubble now displays the configuration of watcher/doer. You may update properties of watcher/doer in the bottom panel. The types of watchers/doers and their details will be discussed in later sections. Make the required changes and click on 'update' on the top right corner of the modifier dialog.



- After this, do not close the speech bubble or unselect the chain in the chain list. Walk to another object that is to be the next object in the chain. Note the chain now attached to your pointer linking it to the previous object. Enter the second object then hold down right mouse button until its speech bubble appears.

- Repeat step 3 with this object. Note that the selected object is already asking you to make it either a watcher or a doer object. You may not create a new interaction chain while already in the middle of creating one. Also you may select more than one of each type.



- Repeat steps 3 & 4 until you have added all desired objects to the chain. Note that the chain is already working as are creating it. Click on 'commit' on any speech bubble of an object that is part of the chain to commit the chain. You may now close the speech bubble.

Note:

- When an object has been set as a watcher or a doer object you will see an icon next to the object indicating which it is as follows.



- These watcher icons will light up when their associated condition is met. You may want to use this to test out that your watcher works as you intended. The doer icon will also light up when the object is performing the action. These icons are useful when viewing the chain from the Bird's eye viewpoint.
- The visitor's avatar may act as only as a doer object and not a watcher. However other watcher object may watch for the avatar.

- Some objects may already have a single object or an avatar-based interaction chain defined, such as a billboard²⁷. In these cases you may adjust its default chain and also click on the checkbox "Make others of my kind like me from now on" to make other objects of this type to be created afterwards assume this new interaction chain(s).
- If you close the speech bubble while the chain incomplete, you may complete it at a later stage by selecting the chain from any member of the list and continue.

Modifying existing interaction chain

To modify an existing chain select the list from any member of the chain via its speech bubble.

After an interaction chain is completed, it is added to a list of each object. This list is accessible in the speech bubble of all objects in the chain.

Note: You may do the following when an item is selected and an expanded item is shown:

- Click on any object in the chain and be taken to that object. Note that the selected object's speech bubble will be displayed in place of the previous one.
- Delete any object's participation from the chain by dragging its name into the bin on the bottom right hand corner. However, this will not delete the actual object.

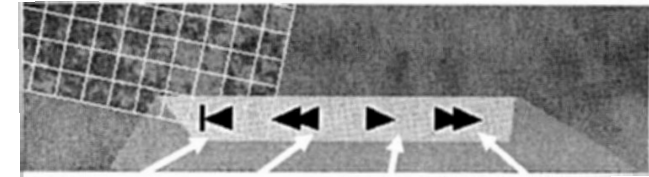
Manipulating Time

At times you may find it necessary to stop, rewind, fast forward or restart time. For example:

²⁷ A Billboard watches out for the visitor's orientation and adjusts its own orientation to face the same way.

- The objects in an interaction chain may be following a global time-based²⁸ sequence of events. In cases like this, it might be necessary to restart the entire timeline from zero.
- The object you would like to select is moving too fast. In this case, you may want to pause the time so you can select it.

To manipulate time you may use the time control located at the bottom right-hand corner of the screen.



Restart Rewind Play/Pause Fast Forward

The available selections are:

Restart: Global time starts from zero. This would affect all local time-based chain as well.

Rewind: Rewind the time by a predetermined amount of time (default is 5 seconds).

Pause/Play: Pause the time. This button will then appear as the play button. Select this to resume time.

Fast forward: fast forward the time by a predetermined amount of time (default is 5 seconds).

Note: There are some restrictions including:

- After a pause, you will be taken back to where you were when you paused the VE.

²⁸ Global time-based sequence of event is based on the global timeline. For example, a watcher may be set to watch for when one minute has elapsed since the virtual environment was run. Compare this to a local time-based sequence of events, e.g. a watcher is set to watch for one minute has elapsed since it has been clicked on.

Saving and Loading VEs

It is important that you regularly save your work. You can do this by accessing the 3D menu under "save and load world". Note the following when saving and loading VEs.

- When you load your VE **for editing**, everything will be exactly the same as it is at the time you saved it, This includes
 - All the interaction chains and the state in which they were in.
 - The position and orientation as you were in when you perform the save operation.
 - The altered default appearance and interaction chains if you have made changes to them. To restore to the original default object, uncheck "Make others of my kind like me from now on"²⁹.
- When you load your VE **for viewing** (as in when your visitor views it), you will appear in starting position³⁰ and orientation that was set during the authoring process. The global time will also start from the beginning.
- The single .veb file that was saved contained all information including skybox and other textures, sound etc. There is no need to copy any other files from any other folders.

Frequently Asked Questions

How do I delete an object and what happens to interaction chains involving it?

To delete an object, access its speech bubble. On it there's a button "Delete me". Click on it to remove this object from the world. However, if this object is involved in one or more interaction chains, you will be given the following options:

Delete this object and all interaction chains involving this object in the VE: All chains that have this object either as a watcher or doer will be deleted.

Delete this object and its references in all interaction chain involving it: Only references to this object in interaction chains are deleted. The chain and other member objects are kept intact. However, this may leave some interaction chain incomplete.

Cancel and return without deleting this object: The object will not be deleted.

How do I set up a starting point an orientation of my visitor?

Move yourself to your required position and orientation. Access the avatar's speech bubble and select "Make this the starting position & orientation".

What do the different states of the "Make others of my kind like me from now on" checkbox mean?

The checkbox "Make others of my kind like me from now on" on speech bubble is intended to help you create many objects of the same type after modifying it once. When you first create a first object of a particular type in your VE, the checkbox will be blank. You may check it and it have a check through it. This indicates that any other object of this type created later will have basically be the clone of this object.

After creating many objects you would have had a population of this type of objects some of which might be the clones of another and some might not be. When you enable their speech bubble, the checkbox will be in various states:

A white blank checkbox: Future objects of this type will follow the original default specification.

A grey blank checkbox with * next to it: Future objects of this type will follow some other specification **not** based on this current object. Clicking on the star will take you the **original** object, providing the current specification.

A white checked checkbox: Future objects of this type will follow the current object's specification. The current object is the original object providing the specification.

A grey checked checkbox with * next to it: This object is a clone of the object currently providing the specification.

What happens to the altered object specification if I delete the object providing the specification?

It will reset back to the original default specification.

²⁹ See FAQ for full detail.

³⁰ See FAQ for how to set this up.

How does the target object drop-down list in the watcher/doer tag work?

An alternative way to specify an interaction involves pre-selecting all or most of the member of the chain. You can do this by selecting '*specify interaction chain*' from the menu. You will then be asked to give a name to this interaction chain or keep the default name. You must now select all objects that are to be part of this new chain; you can do this by walking over and left-clicking on them. Note that as you click on them, each object is *tagged* with a speech bubble with a '?' in place of watcher/doer icons.

After you have *tagged* all the objects you wish to be in the chain, open a speech bubble (by holding down right mouse button) of any of the selected object. You can now follow the usual procedure for adding interaction chains. However in the drop down list of your target objects will be the objects you selected.

How do I 'browse' for a target object?

When creating a new interaction chain, you may need to identify target objects either for a watcher to watch or doer to do something to (e.g. move towards target object). You may do this by clicking browse, and you'll now be able to move around the VE and select the appropriate object.

Appendix B: Survey Invitation E-mails

From: Jakkaphan Tangkuampien (jak@cs.uct.ac.za)
Subject: Online VR Authoring Survey

Dear [Name],

[Name] has indicated to me that you have some experience with VR authoring and that you might be interested in helping me out with my MSc survey on the topic. The survey will be completed online and you can stop and resume the survey at designated point (usually about 5-10 questions apart) at any time. It is anticipated that the survey would take around one hour to complete in total, with a stopping point every 10-15 minutes. At least two months will be available for you to complete this survey and I am anticipating it to start by June.

If you require more detail, please visit <http://people.cs.uct.ac.za/~jtangkua/VEBuilder> for more detail or e-mail me.

If you are interested and would like to help me out, please fill in a sign-up form at <http://people.cs.uct.ac.za/~jtangkua/VEBuilder/signup.html>

Thank you in advance for any help you can give. If you know of anyone who might be in the position to help me with respect to completing the survey, could you please forward this e-mail to that person? Alternatively, you can also send me his e-mail address so I can get in contact with him/her.

Thank you for your time,
Regards,

Jakkaphan Tangkuampien

Collaborative Visual Computing (CVC) Laboratory
Department of Computer Science
University of Cape Town

E-mail : jak@cs.uct.ac.za
Tel : +27 (0) 21 650 2670

From: Jakkaphan Tangkuampien (jak@cs.uct.ac.za)
Subject: Online VR Authoring Survey

Dear [Name],

Thank you for your interest in participating in our survey on VR authoring. Here are some pointers concerning this survey. Please e-mail me if you have any further questions not addressed by the following:

1. You will not be required to enter your name on the survey.
2. You will be able to pause and resume the survey completion process provided that you use the same computer in all sessions and that your browser allows us to do this via the use of cookie. You do not have to worry if this does not make sense, as we will check your computer at the start of the survey.
3. Only completed survey will be used. You will have at least until the end of September to complete the survey.
4. Please make note of or find out your computer's IP address as we will need this information in case of error or failure in a resumption attempt. You will only be requested this information after such an event.
5. If you are ready, please visit <http://people.cs.uct.ac.za/~jtangkua/VEASurvey/> to start completing the survey. Please make sure you have at least 15 minutes for each session.

Thank you for your interest and help
Regards,

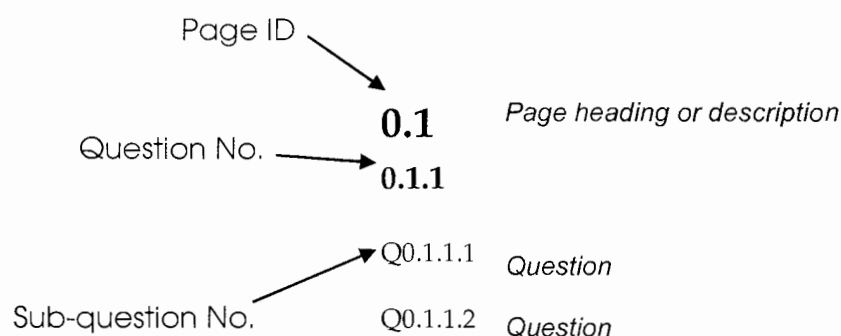
Jakkaphan Tangkuampien

Collaborative Visual Computing (CVC) Laboratory
Department of Computer Science
University of Cape Town

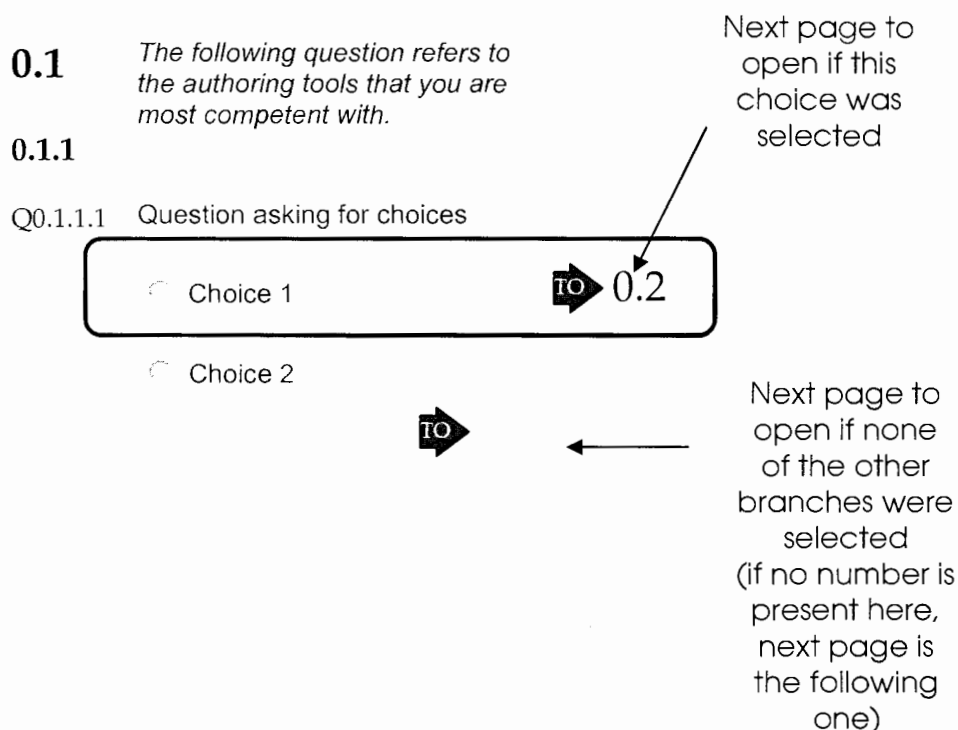
E-mail : jak@cs.uct.ac.za
Tel : +27 (0) 21 650 2670

Appendix C: Online Verification Survey Questions

The survey was conducted online. The questions were originally on an HTML page with validation code to ensure all questions were answered correctly. Pages were then processed by PHP-enabled web server. The questions below were converted to text and the following convention was used. Each box represents an HTML page.



Some boxes represent branches which are taken depending on the answer selected.



If no page number is specified at the bottom of the table, proceed to the next page.

1.1 Personal Details

1.1.1

Q1.1.1.1

Age:

Q1.1.1.2

Occupation:

2.1

The questions on this page concern the platform on which your VE runs (target display platform) and the platform on which you authored/edited your VE. These systems may be the same.

2.1.1

Q2.1.1.1 **Target Display Platform (select all applicable)**

- ☐ Head-mounted Display
- ☐ Single monitor, non-stereo
- ☐ Single monitor with stereo-glasses or other depth-indication methods
- ☐ CAVE systems
- ☐ Other: Specify:

2.1.2 Display System

This is the system where the VE you created was run.

Q2.1.2.1

CPU Speed: Mhz

Q2.1.2.2

Physical Memory: Mb

Q2.1.2.3

Special display accelerator card: (enter NA if not applicable)

Q2.1.2.4

Other components/specification that are of Interest to running your VE:

2.1.3 Editing System

This is the system where you spent the majority of the time creating and editing your VE.

Q2.1.3.1 CPU Speed: Mhz

Q2.1.3.2 Physical Memory: Mb

Q2.1.3.3 Special display accelerator card: (enter NA if not applicable)

Q2.1.3.4 Other components/specification that are of Interest to running your VE:

3.1 The questions on this page refer to the VEs you have created.

3.1.1

Q3.1.1.1 How long have you been creating VEs?
☐ Less than 6 months
☐ 6 month to a year
☐ 1-2 years
☐ Longer than 2 years

3.1.2 Referring to your best VE:

Q3.1.2.1 How important was object placement accuracy?
☐ 1 Didn't Matter ☐ 2 ☐ 3 ☐ 4 Average ☐ 5 ☐ 6 ☐ 7 Extremely Accurate

Q3.1.2.2 What was the realism level?
☐ 1 cartoon like ☐ 2 ☐ 3 ☐ 4 semi-realistic ☐ 5 ☐ 6 ☐ 7 photo realistic

Q3.1.2.3 What was the level of Interactivity?
☐ 1 none ☐ 2 ☐ 3 ☐ 4 some interaction ☐ 5 ☐ 6 ☐ 7 Highly interactive

Q3.1.2.4 What was the frame rate?
☐ 1 <10FPS Very Jerky ☐ 2 ☐ 3 ☐ 4 20-25FPS TV-like ☐ 5 ☐ 6 ☐ 7 >35FPS Hi-speed computer game

Q3.1.2.5 Did the system that run your VE automatically apply gravity for you or did you have to manually simulate it in your environment?:
☐ Yes, the system does it for me

☐ No, I have to manually do it myself

Q3.1.2.6 Did the system that run your VE automatically apply collision detection for you or did you have to manually simulate it in your environment?:

☐ Yes, the system does it for me

☐ No, I have to manually do it myself

Q3.1.2.7

Purpose of the VE:

☐ Architectural

☐ Education

☐ Gaming or Entertainment

☐ Research

☐ Other, specify below:

Q3.1.2.8

Please give a brief description of your VE:

Q3.1.2.9

Rate your VEs in terms of how much it met your initial goals.

☐ 1
Total
Failure

☐ 2

☐ 3

☐ 4
Average

☐ 5

☐ 6

☐ 7
Total
Success

4.1

The questions on this page concern different tasks in VE authoring and the tools which you might have used to perform these tasks.

4.1.1

Generally, creating a VE can be broken down into three areas:

- **Object Modelling:** This involves creating the actual object usually using meshes.
- **Static Environment Building:** This involves adding objects to the empty environment.
- **Interaction Specification:** This involves associating actions with objects and creating events and reactions.

Q4.1.1.1

To what extent do you agree with the above?

☐ 1
Strongly
Disagree

☐ 2

☐ 3

☐ 4
Neutral

☐ 5

☐ 6

☐ 7
Strongly
Agree

4.1.2

Please indicate the authoring tools you normally use in your authoring process, the purpose for its use and your proficiency with the tool. Leave additional fields blank if you have already filled in all the tools you used. Note that we are not asking for all the tools you ever used, but for the tools that you normally use to create a working VE.

A new row will appear once you completed the previous row.

Name of Tool

Purpose (Tick all
appropriate options)

Proficiency

Q4.1.2.1
Q4.1.2.2
Q4.1.2.3

1.

- ☐ Object Modelling
- ☐ Static Environment Building
- ☐ Interaction Specification
- ☐ Others: specify:

Select One

5 Highly Proficient

4

3 Average

2

1 No Experience

Q4.1.2.4
.....

2.

4.1.3

Complete this section only if you use a programming/scripting language to specify interactions.

☐ Tick this box if this is not the case and move on to the next section

Please list the languages you normally use in your authoring process and your proficiency with it. Note that we are not asking for all the programming languages you ever use, just the one s you normally use to create a working VE

A new row will appear once you completed the previous row.

Name of Language

Proficiency

Q4.1.3.1
Q4.1.3.2

1.

Select One ▼

5 Highly Proficient

4

3 Average

2

1 No Experience

Q4.1.3.3
.....

2.

4.2

The following questions concern your authoring process with the tools you have indicated in the previous page.

4.2.1

Q4.2.1.1

How long before you feel that you have mastered the authoring tool(s) you use i.e. when you feel that you were able to create effective VEs in a reasonable amount of time?

- ☐ Less than 1 weeks
- ☐ Between 1 and 3 weeks
- ☐ Between 3 and 6 weeks
- ☐ Longer than 6 weeks

Q4.2.1.2

Base on your experience with your authoring tool(s), how long would it take you to

author a small project, for example, a simple room-based VE with an avatar (VE user) and two agents who greet your visitor if they approach within a set distance?

- ☐ less than 30 minutes
- ☐ 30min-an hour
- ☐ 1-2 hours
- ☐ 2-5 hours
- ☐ Longer than 5 hours

Q4.2.1.3

How do you feel about this amount of time you normally use to create a VE?

- ☐ Could be faster
- ☐ As fast as I can ever get

Q4.2.1.4

On average, how often do you save your VE while in the middle of an authoring session?

- ☐ every 5 mins
- ☐ 15 mins
- ☐ 30 mins
- ☐ An hour

Q4.2.1.5

Which of the following statement best describe how you normally author your VE?

- ☐ I create the entire or the majority of the static environment first, and then add the interactions.
- ☐ I create the larger objects in the static environment first, and then add each object followed by specifying interactions for that object.
- ☐ I switch between the two tasks frequently depending on the situation.

Q4.2.1.6

While authoring, how accurate is your tool's representation of your VE compared to the end-result

- ☐ exactly the same
- ☐ Looks the same, but there are aspects where it is not.
- ☐ not quite the same, but I can get the idea of what it'd look like
It is nothing like the VE, but I'm still able to get the idea.
- ☐ Totally different

4.2.2

Rate the difficulty level in using your tools to do each of the following tasks.

Q4.2.2.1

Selecting an object

- ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐
- Extremely Easy Average Extremely Difficult Depends on the tool

Q4.2.2.2

Moving an object

- ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐
- Extremely Easy Average Extremely Difficult Depends on the tool

Q4.2.2.3

Accurately placing an object

☐ 1 Extremely Easy
 ☐ 2
 ☐ 3
 ☐ 4 Average
 ☐ 5
 ☐ 6
 ☒ 7 Extremely Difficult
 ☐ Depends on the tool

Q4.2.2.4 Changing an object's physical properties

☐ 1 Extremely Easy
 ☐ 2
 ☐ 3
 ☐ 4 Average
 ☐ 5
 ☐ 6
 ☒ 7 Extremely Difficult
 ☐ Depends on the tool

Q4.2.2.5 Add interactions between two or more objects

☒ 1 Extremely Easy
 ☒ 2
 ☒ 3
 ☒ 4 Average
 ☒ 5
 ☒ 6
 ☒ 7 Extremely Difficult
 ☒ Depends on the tool

Q4.2.2.6 Find objects in an interaction (For example, finding a switch that will turn the selected light on, or vice versa)

☐ 1 Extremely Easy
 ☐ 2
 ☐ 3
 ☐ 4 Average
 ☐ 5
 ☐ 6
 ☒ 7 Extremely Difficult
 ☐ Depends on the tool

Q4.2.2.7 Change global setting such as location, weather or time of day

☐ 1 Extremely Easy
 ☐ 2
 ☐ 3
 ☐ 4 Average
 ☐ 5
 ☐ 6
 ☒ 7 Extremely Difficult
 ☐ Depends on the tool

Q4.2.2.8 Create a time-based sequence of events

☐ 1 Extremely Easy
 ☐ 2
 ☐ 3
 ☐ 4 Average
 ☐ 5
 ☐ 6
 ☒ 7 Extremely Difficult
 ☐ Depends on the tool

5.1 The following question refers to the authoring tools that you are most competent with.

Q5.1.1.1 Considering only the static environment and interaction specification tasks, which of these statements best describe the VE authoring tool(s) that you used to create your best VE:

☐ I used only one tool, with two modes; one for each task.

TO 5.2

☐ I used one tool, which allowed me to do either of these tasks without changing modes.

TO 6.1

☐ I used two tools, each performing the different tasks

TO 5.2

5.2 You've indicated that you used two tools/modes while authoring your VE; one for static environment building and the other for interaction specification.

- 5.2.1

To what extent do you agree with the following statements? (You may make any comments concerning each.)
- Q5.2.1.1

There were major incompatibility problems between the different tools.

1

Strongly Disagree

2

3

4

Neutral

5

6

7

Strongly Agree

Comment:
- Q5.2.1.2

The two tools/modes were totally independent. They were little or no integration at all.

1

Strongly Disagree

2

3

4

Neutral

5

6

7

Strongly Agree

Comment:
- Q5.2.1.3

Using separate tools/modes forces me to perform a large chunk of environment building first before I can specify any interaction

1

Strongly Disagree

2

3

4

Neutral

5

6

7

Strongly Agree

Comment:
- Q5.2.1.4

I sometimes get confused about which tool I am using and for what purpose.

1

Strongly Disagree

2

3

4

Neutral

5

6

7

Strongly Agree

Comment:
- Q5.2.1.5

The tool forces me to back up my VE every time I switch between tools or modes.

1

Strongly Disagree

2

3

4

Neutral

5

6

7

Strongly Agree

Comment:

5.3

The questions on this page refer to the issue of using two different tools or modes (for static environmental building and interaction specification).

- 5.3.1

Please indicate the extent to which this issue was problematic when you were learning how to create a VE.

Q5.3.1.1

1

Not a Problem

2

3

4

Somewhat Problematic

5

6

7

Extremely Problematic

TO

5.4

- Q5.3.1.2

Please indicate the extent to which this issue was problematic to your VE authoring process.

1

Not a Problem

2

3

4

Somewhat Problematic

5

6

7

Extremely Problematic

TO

5.4

- Q5.3.1.3

Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.

1

No Effect

2

3

4

Significant Effect

5

6

7

Extremely Negative Effect

TO

5.4

- Q5.3.1.4

Additional comments (negative or positive) on how this issue affected your VE authoring process. You may leave this blank.

5.4

The issue in question: using two different tools or modes (for static environmental building and interaction specification).

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- One single tools with one interface
- All editing steps are atomic and independent of each other, this will lead to compatibility between all tasks

This means that as an author you will be able to use a single tool for building your environment.

5.4.1

Q5.4.1.1

How much do you think the above solution will alleviate the issue?

1

No Effect

2

3

4

Significant Effect

5

6

7

Very Positive Effect

I don't understand your solution

Q5.4.1.2

Suggestions on how this issue could be further resolved. You may leave this blank.

Q5.4.1.3

Additional comments (negative or positive) on this solution. You may leave this blank.

6.1

The following question refers to the authoring tools that you are most competent with.

6.1.1

Q6.1.1.1 Which of these statements most accurately describe the process you are required to go through to preview your final VE product (in all aspects, including interactions), after or during the editing process?

☐ My tool requires that I build or compile my VE or some part of it first before I can preview and fully interact with all aspects of it. TO 6.2

☐ My tool does not require me to build or compile my VE, but it needs to be viewed by an external program that display the VE i.e. I first build the VE in one and then open in another program or mode. TO 6.2

☐ My tool shows the resultant VE complete with interactions, while I am creating it. There's no need to compile or start a separate process, application or mode. TO 7.1

6.2 You've indicated that at least one of the tools you use requires a compilation or a build step before the result can be seen.

6.2.1 Please tick all applicable tasks to which this issue applied.

Q6.2.1.1 ☐ Object Modelling

☐ Static Environment Building

☐ Interaction Specification

6.2.2 To what extent do you agree with the following statements, considering those tools which require a compilation/build step? (You may make any comments concerning each.)

Q6.2.2.1 *Not being able to see an immediate result, I usually try to compile/build as often as I can.*

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 Comment:

Strongly Disagree Neutral Strongly Agree

Q6.2.2.2 *The compilation/build step takes so long that I usually try to do a lot of things before I compile/build.*

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 Comment:

Strongly Disagree Neutral Strongly Agree

Q6.2.2.3 *After each compilation/build, I have to wait a long time to verify a time-based event (such as music play 3 minutes after VE starts), resulting in delay in authoring.*

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 Comment:

Strongly Disagree Neutral Strongly Agree

Q6.2.2.4 *The end-product after compilation/build is so different to what the tool shows me on the screen (before compilation/build) that I spend a lot of time imagining what the result will be like.*

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 Comment:

Strongly
Disagree

Neutral

Strongly
Agree

Q6.2.2.5

There are major differences in the interface that is in use when editing and the one used when viewing the VE.

1
Strongly
Disagree

2

3

4
Neutral

5

6

7
Strongly
Agree

Comment:

6.3

The questions on this page refer to the issue of the tools you use requires that a compilation or a build step before the result can be seen.

6.3.1

Please indicate the extent to which this issue was problematic when you were learning how to create a VE.

Q6.3.1.1

1
Not a
Problem

2

3

4
Somewhat
Problematic

5

6

7
Extremely
Problematic

TO 6.4

Please indicate the extent to which this issue was problematic to your VE authoring process.

Q6.3.1.2

1
Not a
Problem

2

3

4
Somewhat
Problematic

5

6

7
Extremely
Problematic

TO 6.4

Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.

Q6.3.1.3

1
No Effect

2

3

4
Significant
Effect

5

6

7
Extremely
Negative
Effect

TO 6.4

Additional comments (negative or positive) on how this issue affected your VE authoring process. You may leave this blank.

Q6.3.1.4

TO 7.1

6.4

The issue in question: The tools you use require that a compilation or a build step before the result can be seen.

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- WYSIWYG Interface
- Combined edit and view into one, with both tasks accessible at any time during authoring

This means that the author will be able to make changes to the environment and have the changes reflected directly afterwards without having to re-compile or re-build.

6.4.1

Q6.4.1.1 How much do you think the above solution will alleviate the issue?

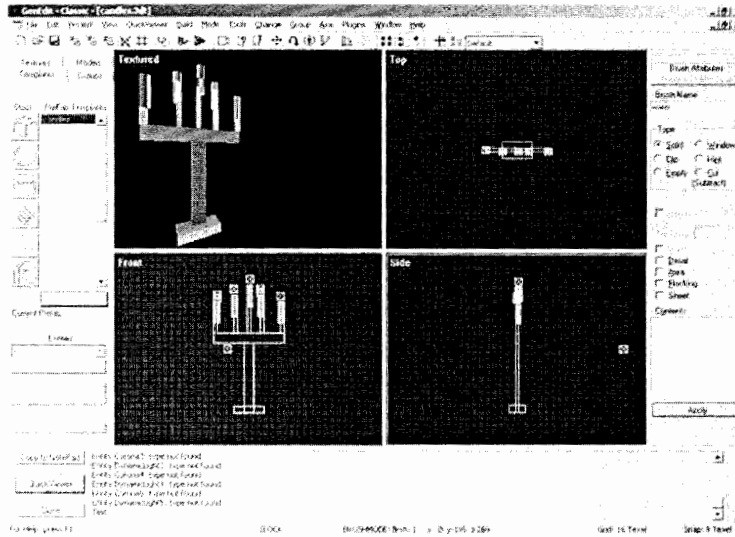
☐ -3	☐ -2	☐ -1	☐ 0	☐ 1	☐ 2	☐ 3	☐
High Negative Effect			No Effect			High Positive Effect	I don't understand your solution

Q6.4.1.2 Suggestions on how this issue could be further resolved. You may leave this blank.

Q6.4.1.3 Additional comments (negative or positive) on this solution. You may leave this blank.

7.1

The image below is a screenshot from a VE authoring tool *GenEdit*. The user may place, move or select objects using the four-pane interface, with the panes showing the camera, top, front, and side views.



7.1.1 Do any of the authoring tools you used have a similar interface for positioning or selecting objects?
Q7.1.1.1

☐ Yes, it does have 3 orthogonal viewpoints and a camera viewpoint.

TO 7.2

☐ Yes, it does have 3 orthogonal viewpoints but **no** camera viewpoint.

TO 7.2

☐ No, but it does have multiple panes showing multiple viewpoints.

TO 7.2

☐ No, there is only one pane showing only one viewpoint at any time.

TO 8.1

7.2

You've indicated that at least one of the tools you use has a 3+ viewpoints interface for placing or selecting objects.

7.2.1 Please tick all applicable tasks for which this issue applied.

Q7.2.1.1

- ☐ Object Modelling
- ☐ Static Environment Building
- ☐ Interaction Specification

7.2.2

I found this interface for placing or selecting objects confusing and overly complex.

Q7.2.2.1

☐ Strongly Disagree 2 3 4 5 6 7 Strongly Agree Comment:

Q7.2.2.2

Most of the time, I do not require the extremely high precision afforded to me by this interface.

☐ 1 Strongly Disagree ☐ 2 ☐ 3 ☐ 4 Neutral ☐ 5 ☐ 6 ☐ 7 Strongly Agree Comment:

Q7.2.2.3 *This interface wastes screen space which could have been better used for something else.*

☐ 1 Strongly Disagree ☐ 2 ☐ 3 ☐ 4 Neutral ☐ 5 ☐ 6 ☐ 7 Strongly Agree Comment:

Q7.2.2.4 *I have trouble placing and/or locating objects, mainly due to the fact that I have to browse through all the different viewpoints.*

☐ 1 Strongly Disagree ☐ 2 ☐ 3 ☐ 4 Neutral ☐ 5 ☐ 6 ☐ 7 Strongly Agree Comment:

7.3 The questions on this page refer to the issue of some of the VE authoring tools have a 3+ viewpoints interface for placing or selecting objects.

7.3.1

Q7.3.1.1 Please indicate the extent to which this issue was problematic when you were learning how to create a VE.

☐ 1 Not a Problem ☐ 2 ☐ 3 ☐ 4 Somewhat Problematic ☐ 5 ☐ 6 ☐ 7 Extremely Problematic **TO 7.4**

Q7.3.1.2 Please indicate the extent to which this issue was problematic to your VE authoring process.

☐ 1 Not a Problem ☐ 2 ☐ 3 ☐ 4 Somewhat Problematic ☐ 5 ☐ 6 ☐ 7 Extremely Problematic **TO 7.4**

Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.

☐ 1 No Effect ☐ 2 ☐ 3 ☐ 4 Significant Effect ☐ 5 ☐ 6 ☐ 7 Extremely Negative Effect **TO 7.4**

7.3.1.4 Additional comments (negative or positive) on how this issue affected your VE authoring process. You may leave this blank.

TO 8.1

7.4 The issue in question: some of the VE authoring tools have a 3+ viewpoints interface for placing or selecting objects.

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- Two interchangeable viewpoints, top view and first person. View can be switched at any time.
- Object can be dropped into the environment directly and reacts similar to the real world in that objects fall to the ground and can not go into each other.

This means that the author can place objects either from the top or by walking around the environment itself (as a visitor would) and place objects where they want.

7.4.1

Q7.4.1.1 How much do you think the above solution will alleviate the issue?

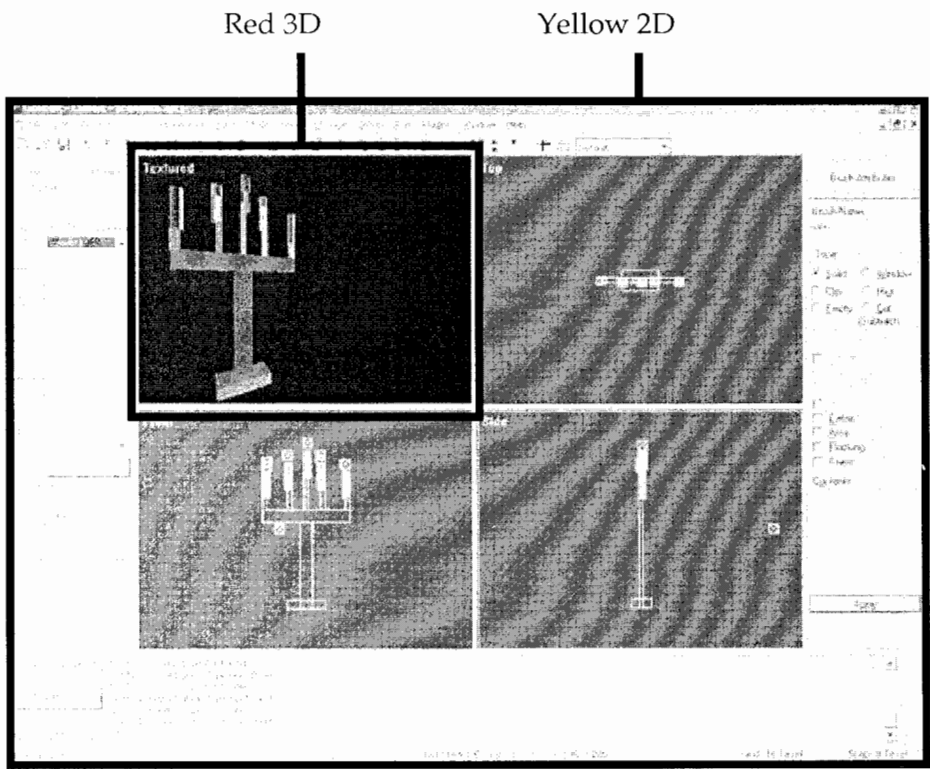


Q7.4.1.2 Suggestions on how this issue could be further resolved. You may leave this blank.

Q7.4.1.3 Additional comments (negative or positive) on this solution. You may leave this blank.

8.1 Here's the same screen captured image from *GenEdit*. The tool has two sets of interfaces indicated by the colours red and yellow. The red region covers the pane where the VE is seen almost like it would be in the end product. The interface in this region is 3D in nature.

The region in yellow is outside of the VE and usually consists of 2D interface. Note that the top, front and side views may have a 3D look but it is in effect a 2D interface since it only allows the user to move objects along a flat plane. *(The label has been added here for clarity and does not appear in the actual questionnaire)*



8.1.1 Do any of the tools have similar mixture of interface types, in and out of the VE being edited?
Q8.1.1.1

- ☐ Yes, the tool I use has only a 2D interface. TO 8.2
- ☐ Yes, the tool I use has mostly 2D interface but it does have some 3D, like *GenEdit* above. TO 8.2
- ☐ Yes, but this is very little, the majority of the interface is 3D TO 8.2
- ☐ No, the tool I use has no 2D interface at all. All of the interface is 3D. TO 9.1

8.2 You've indicated that at least one of the tools you use has a mixture of 2D and 3D interfaces (or only 2D).

8.2.1

Q8.2.1.1 Please tick all applicable tasks for which this issue applied.

- ☐ Object
- ☐ Static Environment
- ☐ Interaction
- ☐ Modelling Building Specification

8.2.2 To what extent do you agree with the following statements which concern 2D and 3D interfaces? (You may make any comments concerning each.)

Q8.2.2.1 *I have to constantly change my mindset from 2D to 3D and vice versa when I'm creating my environment*

☐ 1

☐ 2

☐ 3

☐ 4

☐ 5

☐ 6

☐ 7

Strongly DisagreeNeutralStrongly Agree

Comment:

Q8.2.2.2 *I feel that there are way too many 2D interface (menu, buttons, etc), so much so that it is cluttering up my work area.*

☐ 1

☐ 2

☐ 3

☐ 4

☐ 5

☐ 6

☐ 7

Strongly DisagreeNeutralStrongly Agree

Comment:

Q8.2.2.3 *There is way too much area dedicated to 2D interfaces. I'd rather see more space allocated to editing or viewing my VE.*

☐ 1

☐ 2

☐ 3

☐ 4

☐ 5

☐ 6

☐ 7

Strongly DisagreeNeutralStrongly Agree

Comment:

Q8.2.2.4 *The cursors (or pointers) for each (2D and 3D) set of interface are so different that I sometimes lose track of the pointer.*

☐ 1

☐ 2

☐ 3

☐ 4

☐ 5

☐ 6

☐ 7

Strongly DisagreeNeutralStrongly Agree

Comment:

Q8.2.2.5 *I think of the 2D as the editing portion and the 3D portion as the viewing window where little of the editing process can be done.*

☐ 1

☐ 2

☐ 3

☐ 4

☐ 5

☐ 6

☐ 7

Strongly DisagreeNeutralStrongly Agree

Comment:

Q8.2.2.6 *There is sometimes a long delay before the changes (such as an object's properties), I specify in the 2D portion before I can see it in the 3D portion.*

☐ 1

☐ 2

☐ 3

☐ 4

☐ 5

☐ 6

☐ 7

Strongly DisagreeNeutralStrongly Agree

Comment:

8.3 The questions on this page refer to the issue of the mixture of 2D and 3D interfaces in some VE authoring tools.

8.3.1

Q8.3.1.1 Please indicate the extent to which this issue was problematic when you were learning how to create a VE.

☐ 1 Not a Problem ☐ 2 ☐ 3 ☐ 4 Somewhat Problematic ☐ 5 ☐ 6 ☐ 7 Extremely Problematic **TO 8.4**

Q8.3.1.2 Please indicate the extent to which this issue was problematic to your VE authoring process.

☐ 1 Not a Problem ☐ 2 ☐ 3 ☐ 4 Somewhat Problematic ☐ 5 ☐ 6 ☐ 7 Extremely Problematic **TO 8.4**

Q8.3.1.3 Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.

☐ 1 No Effect ☐ 2 ☐ 3 ☐ 4 Significant Effect ☐ 5 ☐ 6 ☐ 7 Extremely Negative Effect **TO 8.4**

Q8.3.1.4 Additional comments (negative or positive) on how this issue affected your VE authoring process. You may leave this blank.

TO 9.1

8.4 The issue in question: mixture of 2D and 3D interfaces in some VE authoring tools

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- Only one tool with one window displaying the virtual environment. All interfaces from pointer, menu and dialog boxes all appear inside the environment.
- Speech bubble interface to replace dialog boxes for editing of object's properties and interactions with other objects.

This means that the author will be able to walk around the VE and call up any widget, menu, object properties etc and they will appear within the environment itself.

8.4.1

Q8.4.1.1 How much do you think the above solution will alleviate the issue?

☐ -3 High Negative Effect ☐ -2 ☐ -1 ☐ 0 No Effect ☐ 1 ☐ 2 ☐ 3 High Positive Effect ☐ I don't understand your solution

Q8.4.1.2 Suggestions on how this issue could be further resolved. You may leave this blank.

Q8.4.1.3

Additional comments (negative or positive) on this solution. You may leave this blank.

9.1

9.1.1

When selecting an object to edit or to specify interaction, which of these statements best describe this aspect of your tools?

Q9.1.1.1

☐ I have to type in the object's name that I am looking for (Select this option if you use specify interaction via a scripting language and have to type in something like `bigbox.move()`)

TO 9.2

☐ I have to use a drop-down list (or other 2D list interface) to select the object's name among all objects in my VE. (Select this option if you use specify interaction via a scripting language, but your ide/editor have autocomplete feature that shows you all available objects)

TO 9.2

☐ I can just click on the object in 3D view

TO 10.1

9.2

You've indicated that you have to select an object using a list or by typing its name.

9.2.1

Q9.2.1.1

Please tick all applicable tasks for which this issue applied.

- ☐ Object Modelling
- ☐ Static Environment Building
- ☐ Interaction Specification

9.2.2

To what extent do you agree with the following statements concerning this issue? (You may make any comments concerning each.)

Q9.2.2.1

I have difficulty in locating or selecting objects using this method.

☐ 1

☐ 2

☐ 3

☐ 4

☐ 5

☐ 6

☐ 7

Strongly Disagree

Neutral

Strongly Agree

Comment:

Q9.2.2.2

I sometimes take a long time to find the right object because I could not remember the object's name

☐ 1 Strongly Disagree ☐ 2 ☐ 3 ☐ 4 Neutral ☐ 5 ☐ 6 ☐ 7 Strongly Agree Comment:

Q9.2.2.3 *I sometimes get confused between which object I am trying to select and the names which appear on the list (or which I have to type in) since there are no direct links between the two.*

☐ Strongly Disagree ☐ 2 ☐ 3 ☐ 4 Neutral ☐ 5 ☐ 6 ☐ 7 Strongly Agree Comment:

Q9.2.2.4 *I found it difficult to keep track of which object was currently selected*

☐ Strongly Disagree ☐ 2 ☐ 3 ☐ 4 Neutral ☐ 5 ☐ 6 ☐ 7 Strongly Agree Comment:

9.3 The questions on this page refer to the issue of selecting objects by using a list-based interface or typing in their names.

9.3.1

Q9.3.1.1 Please indicate the extent to which this issue was problematic when you were learning how to create a VE.

☐ 1 Not a Problem ☐ 2 ☐ 3 ☐ 4 Somewhat Problematic ☐ 5 ☐ 6 ☐ 7 Extremely Problematic TO 9.4

Q9.3.1.2 Please indicate the extent to which this issue was problematic to your VE authoring process.

☐ 1 Not a Problem ☐ 2 ☐ 3 ☐ 4 Somewhat Problematic ☐ 5 ☐ 6 ☐ 7 Extremely Problematic TO 9.4

Q9.3.1.3 Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.

☐ 1 No Effect ☐ 2 ☐ 3 ☐ 4 Significant Effect ☐ 5 ☐ 6 ☐ 7 Extremely Negative Effect TO 9.4

Q9.3.1.4 Please give any additional comments (negative or positive) on how this issue affected your VE authoring process

TO 10.1

9.4 The issue in question: selecting objects by using a list-based interface or typing in their

names.

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

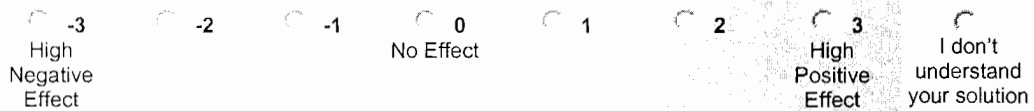
- Direct Object Selection – In conjunction with
 - WYSIWYG Interface where the author walks around the VE in the same way as a visitor might
 - In-Environment interface where widgets (such as menus) appear in the environment (as 3D objects)

This means that authors are able to walk around the VE and reach out to select an object to edit.

9.4.1

Q9.4.1.1

How much do you think the above solution will alleviate the issue?



Q9.4.1.2

Suggestions on how this issue could be further resolved. You may leave this blank.

Q9.4.1.3

Additional comments (negative or positive) on this solution. You may leave this blank.

10.1

10.1.1

Q10.1.1.1

Consider two objects in a VE which are *interactively linked*. For example, a switch may be linked to a light bulb in such a way that if the switch is turned on, the light bulb lights up. Which one of these statements most accurately describes the way your VE authoring tool display such links?

☐ Objects are linked via a programming language but there is not visual way to tell if two objects are linked

TO 10.2

☐ Objects are linked via a programming language and there is some sort of visual link between object (e.g. they are both highlighted), but the only way to tell that they are linked is via the programming language.

TO 10.2

☐ Objects are visually linked and there is a visual representation of their relationship.

TO 11.1

10.2

You've indicated that your authoring tool indicates inter-actional relationships between different objects via a programming language and that there are little or no visual links between them.

10.2.1

To what extent do you agree with the following statements concerning this issue. (You may make any comments concerning each.)

Q10.2.1.1

After selecting an object which is part of an interaction chain or pair, I have trouble locating the other object as there are no visual link between these objects

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7
Strongly Disagree Neutral Strongly Agree

Comment:

Q10.2.1.2

I have difficulty in figuring out the interaction in the chain/pair (i.e. how the objects are acting and responding to actions of each other).

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7
Strongly Disagree Neutral Strongly Agree

Comment:

Q10.2.1.3

Sometimes I spend a long time locating all the objects in the interaction chain/pair in order to change their behaviours. This often takes significantly longer than making the change itself.

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7
Strongly Disagree Neutral Strongly Agree

Comment:

10.3

The issue in question: your authoring tool indicates inter-actional relationships between different objects via a programming language and that there are little or no visual links between them.

10.3.1

Please indicate the extent to which this issue was problematic when you were

Q10.3.1.1

learning how to create a VE.

1

Not a Problem

2

3

4

Somewhat Problematic

5

6

7

Extremely Problematic

TO 10.4

Q10.3.1.2

Please indicate the extent to which this issue was problematic to your VE authoring process.

1

Not a Problem

2

3

4

Somewhat Problematic

5

6

7

Extremely Problematic

TO 10.4

Q10.3.1.3

Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.

1

No Effect

2

3

4

Significant Effect

5

6

7

Extremely Negative Effect

TO 10.4

Q10.3.1.4

Please give any additional comments (negative or positive) on how this issue affected your VE authoring process

TO 11.1

10.4

The issue in question:

your authoring tool indicates inter-actional relationships between different objects via a programming language and that there are little or no visual links between them.

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

Visual Display of this relationship in top-view and first-person view

This means that as an author, you will be able to select and object and have their interaction link shown visually from the top where you'll be able to see the entire relationship.

10.4.1

Q10.4.1.1

How much do you think the above solution will alleviate the issue?

-3

High Negative Effect

-2

-1

0

No Effect

1

2

3

High Positive Effect

I don't understand your solution

Q10.4.1.2

Suggestions on how this issue could be further resolved. You may leave this blank.

Q10.4.1.3

Additional comments (negative or positive) on this solution. You may leave this blank.

11.1

11.1.1

Q11.1.1.1

Often static environment building tools provide some sort of object library from which users can readily add objects to their world. Which statements best describe this facility in your tools?

☐ The tool I use provides no internal library and no way of even importing basic 3D meshes from anywhere. I have to create new object from scratch within the tool

TO 11.2

☐ The tool I use provides no internal library but I can import basic 3D meshes from elsewhere.

TO 11.2

☐ The tool I use provides an internal library containing basic 3D meshes. I can also import or add new objects to the library.

TO 11.2

☐ The tool I use provides an internal library containing 3D objects which also have behaviours attached to them (and not just basic 3D meshes). I can also import or add new objects to the library.

TO 12.1

11.2

11.2.1

You've indicated your authoring tool provides little or no library facility for storing and adding advanced objects (i.e. those with meshes and behaviour already defined).

To what extent do you agree with the following statements concerning this issue. (You may make any comments concerning each.)

Q11.2.1.1

I have to define behaviours for objects that should already have that behaviour defined eg. A door should already have an 'open' action but I have to do this myself.

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☒ 7
Strongly Disagree Neutral Strongly Agree

Comment:

Q11.2.1.2

I have to repeatedly define behaviours for objects which are the same or very similar to those that I have done before.

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☒ 7
Strongly Disagree Neutral Strongly Agree

Comment:

11.3 The questions on this page refer to the issue of authoring tools providing little or no library facility for storing and adding advanced objects (i.e. those with meshes and behaviour already defined).

11.3.1

Q11.3.1.1 Please indicate the extent to which this issue was problematic when you were learning how to create a VE.

1

2

3

4

5

6

7

Not a Problem

Somewhat Problematic

Extremely Problematic

TO 11.4

Q11.3.1.2 Please indicate the extent to which this issue was problematic to your VE authoring process.

1

2

3

4

5

6

7

Not a Problem

Somewhat Problematic

Extremely Problematic

TO 11.4

Q11.3.1.3 Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.

1

2

3

4

5

6

7

No Effect

Significant Effect

Extremely Negative Effect

TO 11.4

Q11.3.1.4 Please give any additional comments (negative or positive) on how this issue affected your VE authoring process

TO 12.1

11.4 **The issue in question:** authoring tools providing little or no library facility for storing and adding advanced objects (i.e. those with meshes/model and behaviours and reaction already defined).

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- Library of objects with meshes/models and appropriate behaviours and reactions

This means that as an author, you will be able to select an object from the library and add it to your VE. The object will also contain behaviours and available actions making these accessible to you with little or no programming.

11.4.1

Q11.4.1.1 How much do you think the above solution will alleviate the issue?

-3

-2

-1

0

1

2

3

High Negative Effect

No Effect

High Positive Effect

I don't understand your solution

Suggestions on how this issue could be further resolved. You may leave this blank.

Q11.4.1.2

Additional comments (negative or positive) on this solution. You may leave this blank.

Q11.4.1.3

12.1 Some authoring tool provides gravity and collision detection facilities **during** editing.

Gravity is provided such that when a new object is dropped mid-air, it would automatically fall the the ground, and that gravity also affects any action or behaviours specified to the object. If the tool does not provide this facility, authors would have to simulate this by adding a gravitational factor to all interaction specified. This is sometime the case with collision detection; authors would have to make sure that objects are not placed inside floors, walls or other objects.

12.1.1 During your static environment building and interaction specification, which of these statements most accurately describes your tool behaviour during the VE authoring process?

Q12.1.1.1

☐ The tool I use provides no gravity and no collision detection.

TO 12.2

☐ The tool I use provides only gravity but no collision detection.

TO 12.2

☐ The tool I use provides no gravity but provides collision detection.

TO 12.2

☐ The tool I use provides both gravity and collision detection.

TO 13.1

12.2 You've indicated your static environment building tool provides either gravity or collision detection facilities or neither.

12.2.1 To what extent do you agree with the following statements concerning this issue. (You may make any comments concerning each.)

Q12.2.1.1 *I have to make sure that objects do not goes through wall or other objects*

☐ 1
Strongly
Disagree

☐ 2

☐ 3

☐ 4
Neutral

☐ 5

☐ 6

☐ 7
Strongly
Agree

Comment:

Q12.2.1.2 *I have to manually placed object on the floor, making sure that it doesn't go through the floor or float in the air.*

☐ 1
Strongly
Disagree

☐ 2

☐ 3

☐ 4
Neutral

☐ 5

☐ 6

☐ 7
Strongly
Agree

Comment:

12.3

The questions on this page refer to the issue of the static environment building and interaction specification tools providing either gravity or collision detection facilities, or neither.

12.3.1

Q12.3.1.1

Please indicate the extent to which this issue was problematic when you were learning how to create a VE.

☐ 1
Not a
Problem

☐ 2

☐ 3

☐ 4
Somewhat
Problematic

☐ 5

☐ 6

☐ 7
Extremely
Problematic

TO 12.4

Q12.3.1.2

Please indicate the extent to which this issue was problematic to your VE authoring process.

☐ 1
Not a
Problem

☐ 2

☐ 3

☐ 4
Somewhat
Problematic

☐ 5

☐ 6

☐ 7
Extremely
Problematic

TO 12.4

Q12.3.1.3

Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.

☐ 1
No Effect

☐ 2

☐ 3

☐ 4
Significant
Effect

☐ 5

☐ 6

☐ 7
Extremely
Negative
Effect

12.4

Q12.3.1.4

Please give any additional comments (negative or positive) on how this issue affected your VE authoring process

TO 13.1

12.4

The issue in question: static environment building and interaction specification tools providing either gravity or collision detection facilities, or neither.

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- Addition of gravity and collision detection facilities
- Abilities to switch these off on a global or per-object basis

This means that as an author, you will be able to easily place objects into VE without worry about them going through walls or other objects. You will also be able to turn these facilities off should the need arises.

12.4.1

Q12.4.1.1

How much do you think the above solution will alleviate the issue?

☐ -3
High
Negative
Effect

☐ -2

☐ -1

☐ 0
No Effect

☐ 1

☐ 2

☐ 3
High
Positive
Effect

☐ I don't
understand
your solution

Suggestions on how this issue could be further resolved. You may leave this blank.

Q12.4.1.2

Additional comments (negative or positive) on this solution. You may leave this blank.

Q12.4.1.3

13.1

During your static environment building, a backdrop is often used to create the illusion of the surrounding environment. This is usually done via a skybox or a skydome where pictures are pasted inside these structures which completely surround the VE.

Other globally affecting factors are weather and the time system. This includes specifying how much light shines on the environment and from which direction. Rain drops can also be simulated within the environment by using billboards or other means.

13.1.1

Q13.1.1.1

Which of these statements most accurately describes your tool behaviour in this respect?

☐ There was no easy way for me to create or change the backdrop, I had to manually simulate this. I also had to simulate weather appearance and other global properties. The tool that I used did not provide any way of specifying this.

TO 13.2

☐ The tool I use provided some way to easily create or change the backdrop but I still had to simulate the weather and other global properties manually.

TO 13.2

☐ The tool I use provided no easy way to create or change the backdrop but it did provide an easy way to specify the weather and/or other global properties manually.

TO 13.2

☐ The tool I used provided an easy way to create or change the backdrop and also provided an easy way to change the weather and other global properties.

TO 14.1

13.2

You've indicated your static environment building tool providing little or no support for easily creating or changing the backdrop, weather and/or other global properties.

13.2.1

To what extent do you agree with the following statements concerning this issue. (You may make any comments concerning each.)

Q13.2.1.1

I have to simulate the backdrop, weather and/or other global properties myself. My tool does not have a way for me to just choose it.

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7
Strongly Disagree Neutral Strongly Agree

Comment:

Q13.2.1.2

The process of creating a backdrop and/or weather is overly complex. There should be an easier way to do this.

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7
Strongly Disagree Neutral Strongly Agree

Comment:

13.3

The questions on this page refer to the issue of the static environment building tool providing little or no support for easily creating or changing the backdrop, weather and/or other global properties.

13.3.1

Q13.3.1.1 Please indicate the extent to which this issue was problematic when you were learning how to create a VE.

1 2 3 4 5 6 7

Not a Problem Somewhat Problematic Extremely Problematic

TO 13.4

Q13.3.1.2 Please indicate the extent to which this issue was problematic to your VE authoring process.

1 2 3 4 5 6 7

Not a Problem Somewhat Problematic Extremely Problematic

TO 13.4

Q13.3.1.3 Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.

1 2 3 4 5 6 7

No Effect Significant Effect Extremely Negative Effect

13.4

Please give any additional comments (negative or positive) on how this issue affected your VE authoring process

TO 14.1

13.4

The issue in question: static environment building tool providing little or no support for easily creating or changing the backdrop, weather and/or other global properties.

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- Ability to change these global properties easily
- Availability of library of settings, time of day and weather.
- In conjunction with the WYSIWYG interface, you will be able to see the results immediately from the point of view of your visitor.

This means that as an author, you will be able to change the location, time of day or weather of VE easily.

13.4.1

Q13.4.1.1 How much do you think the above solution will alleviate the issue?

-3 -2 -1 0 1 2 3

High Negative Effect No Effect High Positive Effect I don't understand your solution

Q13.4.1.2 Suggestions on how this issue could be further resolved. You may leave this blank.

Q13.4.1.3 Additional comments (negative or positive) on this solution. You may leave this blank.

14.1

14.1.1 Concerning adding or specifying interactions to a group of objects, which one of these statements most accurately describes your tool behaviour?

Q14.1.1.1

- ☐

The tool I use requires me to know a programming or scripting language. However, there is no help from the tool at all from this point on. I have to 'program' the interactions myself.

TO 14.2
- ☐

The tool I use requires me to know a programming or scripting language, but it does provide some help by providing access to object's abilities or properties so I do not have to memorise everything myself.

TO 14.2
- ☐

The tool I use requires me to know logical or programming-like structures such as conditional or loop structures. The tool can then take me through the process of specifying interactions without me having to do any programming or scripting. However, there is no other way for me to specify interactions.

TO 15.1
- ☐

The tool I use is similar to the one in the previous option, but I can also program or script the interactions if I choose to.

TO 15.1

14.2 You've indicated that your tool requires you to know a programming/scripting language in order to specify interactions.

14.2.1 To what extent do you agree with the following statements concerning this issue. (You may make any comments concerning each.)

Q14.2.1.1 I have to specifically learn how to a programming or scripting language for this tool.

1

Strongly Disagree

2

3

4

Neutral

5

6

7

Strongly Agree

Comment:

Q14.2.1.2 I find it difficult at times to add interactions or behaviours of objects by programming or scripting.

1

Strongly Disagree

2

3

4

Neutral

5

6

7

Strongly Agree

Comment:

I sometimes avoid adding complex interactions because I found it difficult to specify

194

Q14.2.1.3 *them via programming or scripting*

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7
Strongly Disagree Neutral Strongly Agree

Comment:

Q14.2.1.4

I feel that simple interactions take too long to do via programming or scripting.

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7
Strongly Disagree Neutral Strongly Agree

Comment:

Q14.2.1.5

I spent a lot of time repeatedly programming or scripting similar or slightly varied interactions.

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7
Strongly Disagree Neutral Strongly Agree

Comment:

Q14.2.1.6

I have to do the programming or scripting using a window/tool/editor that seems to be separate from the main tool.

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7
Strongly Disagree Neutral Strongly Agree

Comment:

Q14.2.1.7

There is not enough visual link between the programming or scripting interface (window/tool/editor) and the associated object in the VE. (i.e. when I edit an interaction I find it difficult to tell visually the objects which are taking part)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7
Strongly Disagree Neutral Strongly Agree

Comment:

Q14.2.1.8

The programming/scripting language needs to be compiled before the interaction is added to the VE, making it tedious and time-consuming to arrive at the desired result.

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7
Strongly Disagree Neutral Strongly Agree

Comment:

14.3

The questions on this page refer to the issue of your tool requiring you to know a programming/scripting language in order to specify interactions.

14.3.1

Q14.3.1.1 Please indicate the extent to which this issue was problematic when you were learning how to create a VE.

☐ 1 Not a Problem ☐ 2 ☐ 3 ☐ 4 Somewhat Problematic ☐ 5 ☐ 6 ☐ 7 Extremely Problematic **TO 14.4**

Q14.3.1.2 Please indicate the extent to which this issue was problematic to your VE authoring process.

☐ 1 Not a Problem ☐ 2 ☐ 3 ☐ 4 Somewhat Problematic ☐ 5 ☐ 6 ☐ 7 Extremely Problematic **TO 14.4**

Q14.3.1.3 Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.

☐ 1 No Effect ☐ 2 ☐ 3 ☐ 4 Significant Effect ☐ 5 ☐ 6 ☐ 7 Extremely Negative Effect **TO 14.4**

Please give any additional comments (negative or positive) on how this issue affected your VE authoring process

TO 15.1

14.4

The issue in question: your tool requires you to know a programming/scripting language in order to specify interactions.

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- Specification is done by selecting object directly, telling one to watch for certain events and react by telling others (or itself) to do something.
- Object Library with ready behaviours and reactions for common objects
The system actively prevents impossible action from happening (for example, you might two incompatible actions triggered off by different conditions being watched for).

For example you can tell an actor that:

- When its 3pm, play the flute for 20 minutes
- When someone's at the door, answer it

When its 3pm, the actor will watch TV and any time within the 30 minutes, someone knocks on the door, the actor will stop playing the flute first, and not answer the door while still playing the flute.

This means that as an author, you will be able specify interaction by going to the appropriate objects and tells it to watch and react the way you want. The personification of objects should allow you to easily think about the interactions, while the system actively preventing impossible actions, allowing you to focus on specifying one interaction at a time.

14.4.1

Q14.4.1.1 How much do you think the above solution will alleviate the issue?



Q14.4.1.2 Suggestions on how this issue could be further resolved. You may leave this blank.

Q14.4.1.3 Additional comments (negative or positive) on this solution. You may leave this blank.

15.1

After and sometime, during the processes of static environment building and interaction specification, it is often necessary to *start off* the events rolling so that the often unpredictable result of an interaction chain can be seen. Considering this, which of the following statements best describe how your tool deal with this aspect?

15.1.1

Q15.1.1.1 After interaction specification, the event sequence can be started.

☐ However, I do not have any control over the interaction chain until all interaction are done. **TO** 15.2

☐ There's **no** advanced interface for rewinding, fast forwarding and/or slow motion. However, I can stop the sequence but I'd have to restart the sequence from the beginning again should I make any changes. **TO** 15.2

☐ I can rewind, fast forward, slow down or pause the sequence, though it'd have to be restarted if I choose to make changes during the pause or running of the sequence. **TO** 16.1

☐ I can rewind, fast forward, slow down or pause the sequence, make changes, then able to resume the sequence. **TO** 15.3

15.2

You've indicated your tool does not give you full control of time while running an event sequence and/or that you must restart the sequence if you've made any change.

15.2.1

To what extent do you agree with the following statements concerning this issue. (You may make any comments concerning each.)

Q15.2.1.1 *It is difficult to make sure that simultaneous events take place correctly, because of the lack of the ability to pause time.*



Q15.2.1.2 *I have to spend a lot of time waiting for a sequence to reach the point that I want to see because there's no way to fast forward.*

1

2

3

4

5

6

7

Strongly Disagree

Neutral

Strongly Agree

Comment:

Q15.2.1.3 *At times, I wish I have the ability to pause, fast forward or rewind my VE during the authoring process.*

1

2

3

4

5

6

7

Strongly Disagree

Neutral

Strongly Agree

Comment:

15.3 The questions on this page refer to the issue of the VE authoring tool not giving you full control of time while running an event sequence and/or that you must restart the sequence if you've made any change.

15.3.1

Q15.3.1.1 Please indicate the extent to which this issue was problematic when you were learning how to create a VE.

1

2

3

4

5

6

7

Not a Problem

Somewhat Problematic

Extremely Problematic

TO

15.4

Q15.3.1.2 Please indicate the extent to which this issue was problematic to your VE authoring process.

1

2

3

4

5

6

7

Not a Problem

Somewhat Problematic

Extremely Problematic

TO

15.4

Q15.3.1.3 Please indicate the extent to which this issue has a negative effect on the overall quality of your VE.

1

2

3

4

5

6

7

No Effect

Significant Effect

Extremely Negative Effect

TO

15.4

Q15.3.1.4 Please give any additional comments (negative or positive) on how this issue affected your VE authoring process

TO

16.1

15.4 **The issue in question:** VE authoring tools not giving you full control of time while running an event sequence and/or that you must restart the sequence if you've made any change.

You have indicated that you found this issue to be problematic and/or that it has a negative effect on the overall quality of your VE. We are proposing a solution that might alleviate the issue as follows:

- A VCR-like control with pause, fast forward, rewind and restart.

This means that as an author, you will be able to manipulate time for many purposes such as making sure a sequence of events happen correctly.

15.4.1

Q15.4.1.1 How much do you think the above solution will alleviate the issue?

☐ -3	☐ -2	☐ -1	☐ 0	☐ 1	☐ 2	☐ 3	☐
High Negative Effect			No Effect			High Positive Effect	I don't understand your solution

Q15.4.1.2 Suggestions on how this issue could be further resolved. You may leave this blank.

Q15.4.1.3 Additional comments (negative or positive) on this solution. You may leave this blank.

16.1

16.1.1

Q16.1.1.1

Please indicate any other major problems concerning your virtual authoring tools that were not mentioned during the course of the survey. Please also indicate the extent to which this was problematic to your authoring process. You may skip this question if you did not experience any other major problems.

Q16.1.1.2 Please give suggestions on how the issues you specified above can be resolved.

Bibliography

- [1] Acevedo D, Vote E, Laidlaw D, Joukowsky M – **Archaeological Data Visualization in VR: Analysis of Lamp Finds at the Great Temple of Petra, A Case Study**, *Proceedings of the Conference on Visualization '01*, pp 493-496, 2001
- [2] Ackermann P – **Direct Manipulation of Temporal Structures in a Multimedia Application Framework**, *Proceedings of the second ACM international conference on Multimedia*, San Francisco, California, United States, pp 51 – 58, 1994
- [3] Avison D, Baskerville R, Myers M, Wood-Harper T and Kock N (Chair) – **IS action research: can we serve two masters?**, Panel session, International Conference on Information Systems, *Proceeding of the 20th international conference on Information Systems*, Charlotte, North Carolina, United States - pp 582 - 585, 1999
- [4] Avison D, Lau F, Myers M, and Nielsen PA – **Action Research**, *Communications of the ACM*, Volume 42, Issue 1 (January 1999), pp 94 - 97, 1999
- [5] Babbie E, Mouton J, Vorster P, Prozesky B – **The Practice of Social Research**, South African Edition, Published by Oxford University Press, 2001
- [6] Baskerville RL – **Investigating information systems with action research**, *Communications of the AIS*, Volume 2, Issue 3 (November 1999), Article No. 4, 1999
- [7] Bederson BB – **Fisheye Menus**, *Proceedings of the 13th annual ACM symposium on User interface software and technology*, San Diego, California, United States, pp 217 – 225, 2000
- [8] Best S, Gordon D, Haddad I – **Kernel korner: IBM's Journaled Filesystem**, *Linux Journal*, Issue 105 (January 2003), Page: 9, 2003
- [9] Borning A, Duisberg R – **Constraint-based Tools for Building User Interfaces**, *ACM Transactions on Graphics (TOG)*, Volume 5, Issue 4 (October 1986), Special issue on user interface software, pp 345 – 374, 1986
- [10] Bowman DA, Johnson DB, Hodges LF - **Testbed Evaluation of Virtual Environment Interaction Techniques**, *Proceedings of the ACM symposium on Virtual reality software and technology*, London, United Kingdom, pp 26 - 33, 1999
- [11] Brelsford J – **Physics Education in a Virtual Environment**, *Proceedings of Human Factors and Ergonomics Society, 37th Annual Meeting*, pp 1286-1290, 1993
- [12] Burnett M – **Visual Object-orientated Programming**, *Addendum to the proceedings on Object-oriented programming systems, languages, and applications*, Washington, D.C., United States, pp 127 - 129, 1994

- [13] Callahan J, Hopkins D, Weiser M, Shneiderman B – **An Empirical Comparison of Pie vs. Linear Menus**, *Proceedings of the SIGCHI conference on Human factors in computing systems*, Washington, D.C., United States, pp 95 – 100, 1988
- [14] Cooperrider DL, Srivastava S – **Appreciative Inquiry in Organizational life**, in Woodman R & Pasmore, W. *Research in organizational change and development*, volume 1, Connecticut: JAI-Press, 1987, pp 129-169
- [15] Cruz-Neira C, Sandin DJ, DeFanti, TA, Kenyon R, and Hart JC – **The CAVE, Audio Visual Experience Automatic Virtual Environment**, *Communications of the ACM*, June 1992. pp. 64-72
- [16] Cruz-Neira C, Sandin DJ, DeFanti TA – **Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE**, *ACM Computer Graphics*, Vol. 27, Number 2 , July 1993.pp 135-142.
- [17] Cruz-Neira C – **A Report on Visualization Application in the CAVE Virtual Reality Environment**, *IEEE Symposium on Research Frontiers in Virtual Reality*, Oct. 25-26, San Jose. CA, pp 59-66, 1993
- [18] Dick B – **Action Research: Action and Research** (2002), Available at <http://www.scu.edu.au/schools/gcm/ar/arp/aandr.html>, last accessed 15th January 2005
- [19] Docimo SG, Moore RG, Kavoussi LR – **Telerobotic Surgery in Clinical Reality: Current Experience with Telemonitoring in Adults and Children**, *Presence: Teleoperators and Virtual Environments*, 6(2): pp 173–178, 1997.
- [20] Ehmann S, Lin M – **Accurate Proximity Queries between Convex Polyhedra by Multi-level Voronoi Marching**, *Computer Graphics Forum*, September 2001, vol. 20, no. 3, pp. 500-510, 2001
- [21] Elliott C, **Locating the Energy for Change: An Introduction to Appreciative Inquiry**, published by the International Institute for Sustainable Development, 1999, available : <http://www.iisd.org/publications/publication.asp?pno=287>
- [22] **Encyclopaedia Britannica** – Entry for 'Virtual Reality' – available online: <http://www.britannica.com/eb/article?tocId=9001382&query=Virtual%20Reality&ct=>, last accessed 15 February 2005
- [23] Fencott C – **Towards a Design Methodology for Virtual Environments**, *The International Workshop on User Friendly Design of Virtual Environments*, York, England, 1999
- [24] Frécon E, Greenhalg C and Stenius M – **The DIVE-BONE – An Application-level network architecture for Internet-based CVEs**, *Proceedings of the ACM Virtual Reality Software and Technology (VRST'99)*, London, UK, pp 41-57, 1999

- [25] Govindaraju N, Redon S, Lin M, Manocha D – **COLLIDE: Interactive Collision Detection Between Complex Models in Large Environments using Graphics hardware**, *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, San Diego, California, SESSION: Shading, surfaces, and collision detection, pp 25 - 32 , 2003
- [26] Greenhalgh C, Purbrick J, Benford S, Craven M, Drozd A, Taylor I, **Temporal Links: Recording and Replaying Virtual Environments**, *Proceedings of the eighth ACM international conference on Multimedia*, Marina del Rey, California, United States, pp 67 - 74 , 2000
- [27] Grossman T, Balakrishnan R – **Pointing at Trivariate Targets in 3D Environments**, *Proceedings of the 2004 conference on Human factors in computing systems*, Vienna, Austria, pp 447 - 454 , 2004
- [28] Hall, J – **Strength Based Youth Development: The Banana Kelly Experience**, In S. Hammond & C. Royal (Eds.), *Lessons From the Field – Applying Appreciative Inquiry* (chp. 8). Thin Book Publishing Company, 2001
- [29] Hammond S, Royal C (Eds) – **Lessons From the Field: Applying Appreciative Inquiry (Revised Edition)**, Thin Book Publish Company, 2001
- [30] Hammond S – **The Thin Book of Appreciative Inquiry**, Thin Book Publishing Company, 2nd Edition, 1998
- [31] Hancock MS, Booth KS – **Improving Menu Placement Strategies from Pen Input**, *Proceedings of the 2004 conference on Graphics interface*, Ontario, Canada, pp 221 – 230, 2004
- [32] Hoof K, Zaferakis A, Lin M, Manocha D – **Fast and Simple 2-D Geometric Proximity Queries Using Graphics Hardware**, *Proceedings ACM Symposium on Interactive 3-D Graphics*, pp 145-148, 2001
- [33] Hudson S – **Adding Shadows to a 3D Cursor**, *ACM Transactions on Graphics (TOG)*, Volume 11 , Issue 3 (July 1992), pp 193 - 199, 1992
- [34] Jimenez P, Thomas F, Torras C – **3D Collision Detection: A Survey**, *Computer Graphics* vol 24, pp 269-285, 2001
- [35] Kaur K – **Designing Virtual Environments for Usability**, PhD thesis, Centre for HCI Design, City, University, London, June 1998
- [36] Kobayashi M, Igarashi T – **Considering the Direction of Cursor Movement for Efficient Traversal of Cascading Menus**, *Proceedings of the 16th annual ACM symposium on User interface software and technology*, Vancouver, Canada, pp 91 – 94, 2003
- [37] Leigh J, Johnson A, DeFanti T, Brown M – **A Review of Tele-immersive Applications in the CAVE Research Network**, *Proceedings of the IEEE Virtual Reality (VR'99)*, Houston, United States, pp 180-187, 1999

- [38] Lombard M, Ditton T – **At the Heart of It All: The Concept of Presence**, *Journal of Computer Mediated Communication*, September 1997
- [39] Maes P – **Agents that reduce work and Information Overload**, *Communications of the ACM*, Volume 37 , Issue 7 (July 1994), pp 30 - 40 , 1994
- [40] Maes P, Shneiderman B – **Direct Manipulation vs Interface Agents**, *Interactions*, Volume 4, Issue 6, Nov./Dec. 1997, pp 42 - 61 , 1997
- [41] Masterson TF, Meyer RM – **SIVIL: A True Visual Programming Language for Students**, *Proceedings of the sixth annual CCSC northeastern conference on The journal of computing in small colleges*, Middlebury, Vermont, United States, pp 74 – 86, 2001
- [42] Marsden, G. (1995) – **Overcoming Design and Execute Modes in User Design Environments**, HCI'95 Ancillary proceedings. British Computer Society. Huddersfield
- [43] Massari A, Saladini L – **Virgilio: a VR-based System for Database Visualization**, *Proceedings of the workshop on Advanced Visual Interfaces*, pp 263-265, 1993
- [44] Meyer RM, Masterson T – **Towards a Better Visual Programming Language: Critiqueing PROGRAPH's Control Structures**, *Proceedings of the fifth annual CCSC northeastern conference on The journal of computing in small colleges*, Ramapo College of New Jersey, Mahwah, New Jersey, United States, pp 181 – 193, 2000
- [45] Myers BA – *Languages for developing user interfaces*, A. K. Peters, Ltd., Natick, MA, 1992
- [46] Miyashita K, Matsuoka S, Takahashi S, Yonezawa A, Kamada T – **Declarative Programming of Graphical Interfaces by Visual Examples**, *Proceedings of the 5th annual ACM symposium on User interface software and technology*, Monterey, California, United States, pp 107 - 116 , 1992
- [47] Nielson G, Olsen D – **Direct Manipulation Techniques for 3D Objects using 2D Locator Devices**, *Proceedings of the 1986 workshop on Interactive 3D graphics*, Chapel Hill, North Carolina, United States, pp 175 – 182, 1987
- [48] O'Sullivan C, Dingliana J – **Collisions and Perception**, *ACM Transactions on Graphics (TOG)*, Volume 20 , Issue 3 (July 2001), pp 151 - 168 , 2001
- [49] Obeysekare U, Williams C, Durbin J, Rosenblum L, Rosenberg R, Grinstein F, Ramamurti R, Landsberg A, Sandberg W – **Virtual Workbench - A Non-Immersive Virtual Environment for Visualizing and Interacting with 3D Objects for Scientific Visualization**, *Proceedings of the conference on Visualization '96*, pp 345-349, 1993
- [50] Olivier MS – **Information Technology Research: A practical guide for Computer Science and Informatics**, published by Van Schaik Publishers 2004

- [51] Orne MT – **On the Social Psychology of the Psychological Experiments: With Particular Reference to Demand Characteristics and their Implications**, *Symposium on the Social Psychology of the Psychological Experiment*, American Psychological Association Convention, New York, United States, 1961
- [52] Pierce JS, Forsberg A, Conway MJ, Hong S, Zeleznik R, Mine MR – **Image Plane Interaction Techniques in 3D Immersive Environments**, *Proceedings of the 1997 symposium on Interactive 3D graphics*, Providence, Rhode Island, United States, pp 39 -43, 1997
- [53] Poupyrev I, Billinghurst M, Weghorst S, Ichikawa T – **The Go-Go Interaction Technique: Non-linear Mapping for Direct Manipulation in VR**, *Proceedings of the 9th annual ACM symposium on User interface software and technology*, Seattle, Washington, United States, pp 79 – 80, 1996
- [54] Pradhan R – **Appreciative Inquiry: The Art of Constructing a Positive Future**, Approach Document for UN-HABITAT (United Nations Human Settlements Programme), HIV/Aids Urban Management Programme, 2000. Available: <http://www.unhabitat.org/prograammes/hiv/documents/appreciative1.pdf>
- [55] Pradhan R – **Appreciative Inquiry (AI) Discovery Phase: A Guideline**, Approach Document for UN-HABITAT (United Nations Human Settlements Programme), HIV/Aids Urban Management Programme, 2003. Available: <http://www.unhabitat.org/prograammes/hiv/documents/appreciative2.pdf>
- [56] Pradhan R – **Interviewing Techniques AI**, Approach Document for UN-HABITAT (United Nations Human Settlements Programme), HIV/Aids Urban Management Programme, 2000. Available: http://www.unhabitat.org/prograammes/hiv/documents/magic_power.pdf
- [57] Seo J, Kim GJ – **Explorative Construction of Virtual Worlds: An Interactive Kernel Approach**, *Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*, Singapore, SESSION: 7-3 Modeling, pp 395 - 401 , 2004
- [58] Sherman W, Craig A – **Literacy in Virtual Reality, A New Medium**, *Computer Graphics*, 29(4), pp 37-42, November 1995
- [59] Shneiderman, B – **Direct manipulation: a step beyond programming languages**, *IEEE Computer* 16, 8 (August 1983), 57-69.
- [60] Shneiderman B – **Designing the User Interface: Strategies for Effective Human-Computer Interaction**, Addison-Wesley, 1987
- [61] Shu NC – **Visual Programming**, 1998, Van Nostrand Reinhold Company Inc., New York
- [62] Smith S, Duke D – **Binding Virtual Environments to Toolkit Capabilities**, *Computer Graphics Forum*, 19(3), pp 81–89, 2000

- [63] Steuer J – **Defining Virtual Reality: Dimensions Determining Telepresence**, *Social Responses to Communication Technologies*, pp 1–25, 1993
- [64] Strickland D, Hodge L, North M, Weghorst S – **Overcoming Phobias by Virtual Exposure**, *Communications of the ACM*, Volume 40, No. 8, pp 34-39, August 1997
- [65] Sawaragi T, Horiguchi Y – **Ecological interface enabling human-embodied cognition in mobile robot teleoperation**, *Intelligenc*, Volume 11, Issue 3, pp 20-32, 2000
- [66] Summit P – **A Look at the Construction of Social Reality in Virtual Reality**, Available: <http://www.summittnewmedia.com/Paul/nonfiction/myworld.htm>, 1993-2001, Last accessed: 9 November 2003
- [67] van der Haar D, Hosking DM – **Evaluating Appreciative Inquiry: A Relational Constructionist Perspective**, *Human Relations*, 57, 8, pp 1017-1036, 2004
- [68] Von Schweber E, Von Schweber L – **Virtual Reality: Virtually Here**, *PC Magazine*, Issue #5, March 14, 1995
- [69] Website – **3D Studio Max** – <http://www.3dmax.com> , Last accessed: 19 October 2003
- [70] Website – **Alias Products and Services: Maya** – <http://www.alias.com/eng/products-services/maya/> , Last accessed: 10 November 2003
- [71] Website – **Alice v2.0b - Learn to Program Interactive 3D Graphics**, Carnegie Mellon University – <http://www.alice.org> , last accessed 7 February 2005
- [72] Website – **Appreciative Inquiry Commons** – <http://www.connection.cwru.edu/ai/> , Last accessed: 15 January 2004
- [73] Website – **CsGL – C# Graphics Library** – <http://csgl.sourceforge.net>, Last accessed 25 January 2005
- [74] Website – **Electronic Visualization Lab** – <http://www.evl.uic.edu>, Last accessed: 25 January 2005
- [75] Website – **FloorPlan** – <http://www.floorplan.com>, Last accessed: 9 November 2003
- [76] Website – **Genesis 3D** – <http://www.genesis3d.com>, Last accessed: 19 October 2003
- [77] Website – **HIV/AIDS Quality of Life: Education and Awareness** – http://www.cs.uct.ac.za/Research/CVC/Projects/current/hons_hivVirus – Last accessed: 19 October 2003

- [78] Website – **isee Systems** – <http://iseesystems.com> , last accessed 9 February 2005
- [79] Website – **Merriam-Webster Dictionary Entry for ‘Virtual Reality’** – <http://www.m-w.com/cgi-bin/dictionary?book=Dictionary&va=virtual%20reality> – Last accessed: 19 October 2003
- [80] Website – **National Instruments** – <http://www.ni.com/labview> , last accessed 9 February 2005
- [81] Website – **Python Programming Language** – <http://www.python.org> – Last accessed: 9 November 2003
- [82] Website – **Reiser 4** - <http://www.namesys.com/v4/v4.html> , last accessed 7 February 2005
- [83] Website – **Simulation Laboratories, NASA Ames Research Center** – <http://www.simlabs.arc.nasa.gov/> – Last accessed: 9 November 2003
- [84] Website – **The Importance of a Journaled File System, The Backup Book** - http://www.backupbook.com/03Freezes_and_Crashes/02Journaling.html , last accessed 7 February 2005
- [85] Website – **Virtual Environments Program** - <http://acsys.anu.edu.au/virtual.html> - Last accessed: 11 November 2005
- [86] Website – **Visual Studio Home** – <http://msdn.microsoft.com/vstudio> - Last accessed: 9 November 2003
- [87] Website – **WorldViz’s Vizard VR Toolkit** – <http://www.worldviz.com/vizard/index.html> – Last accessed: 13 November 2005
- [88] Wolber D, **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, United States, pp 117 – 118, 1998
- [89] Yang S, Marsden G – **Using Programming Tools in Virtual Environments**, Technical Report CS02-04-00, Department Of Computer Science, University of Cape Town, 2002. Available: <http://www.cs.uct.ac.za/Research/CVC/Techrep/CS02-04-00.pdf> – Last accessed 3 March 2003
- [90] Zhai S, Buxton W, Milgram P – **The “Silk Cursor”: Investigating Transparency for 3D Target Acquisition**, *Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence*, Boston, Massachusetts, United States, pp 459 – 464, 1994