Intelligence Integration in Distributed Knowledge Management

Dariusz Król
Wroclaw University of Technology, Poland

Ngoc Thanh Nguyen
Wroclaw University of Technology, Poland

Information Science REFERENCE

INFORMATION SCIENCE REFERENCE

Hershey · New York

Director of Editorial Content: Kristin Klinger
Managing Development Editor: Kristin M. Roth
Senior Managing Editor: Jennifer Neidig
Managing Editor: Jamie Snavely
Assistant Managing Editor: Carole Coulson
Copy Editor: Lanette Ehrhardt
Typesetter: Jeff Ash
Cover Design: Lisa Tosheff
Printed at: Yurchak Printing Inc.

Published in the United States of America by
Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue, Suite 200
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com

and in the United Kingdom by
Information Science Reference (an imprint of IGI Global)
3 Henrietta Street
Covent Garden
London WC2E 8LU
Tel: 44 20 7240 0856
Fax: 44 20 7379 0609
Web site: http://www.eurospanbookstore.com

Library of Congress Cataloging-in-Publication Data

Intelligence integration in distributed knowledge management / Dariusz Krol and Ngoc Thanh Nguyen, editors.

p. cm.

Includes bibliographical references and index.

Summary: "This book covers a broad range of intelligence integration approaches in distributed knowledge systems, from Web-based systems through multi-agent and grid systems, ontology management to fuzzy approaches"--Provided by publisher.

ISBN 978-1-59904-576-4 (hardcover) -- ISBN 978-1-59904-578-8 (ebook)

1. Expert systems (Computer science) 2. Intelligent agents (Computer software) 3. Electronic data processing--Distributed processing. I. Krol, Dariusz. II. Nguyên, Ngoc Thanh.

QA76.76.E95I53475 2009

006.3--dc22

2008016377

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Chapter V
A Methodology of Design for Virtual Environments

Clive Fencott
University of Teesside, UK

ABSTRACT

This chapter undertakes a methodological study of virtual environments (VEs), a specific subset of interactive systems. It takes as a central theme the tension between the engineering and aesthetic notions of VE design. First of all method is defined in terms of underlying model, language, process model, and heuristics. The underlying model is characterized as an integration of Interaction Machines and Semiotics with the intention to make the design tension work to the designer's benefit rather than trying to eliminate it. The language is then developed as a juxtaposition of UML and the integration of a range of semiotics-based theories. This leads to a discussion of a process model and the activities that comprise it. The intention throughout is not to build a particular VE design method, but to investigate the methodological concerns and constraints such a method should address.

INTRODUCTION AND PROBLEM STATEMENT

Interactive systems (ISs) are becoming ubiquitous to the extent that there is the very real possibility of their disappearing altogether, at least in the sense of users' perceptions of them as entities worthy of conscious identification. This very ubiquity will largely be the result of effective design, which results in ISs becoming so embedded in our everyday lives that we use them without conscious thought. We can draw an analogy here with the electric motor, which pervades almost all everyday technologies and yet is hardly ever noticed. In the early twentieth century, it was possible to buy electric motors for the home along with a variety of attachments for food preparation, hair drying, vacuum cleaning, and so on. Today we buy specialized gadgets, many of which contain electric motors that go largely unnoticed by us. Even the mobile phone contains an electric motor that is weighted to spin off-centre in order to

create the vibrations that can silently signify an incoming call.

Will this ever be the case with ISs? Will they ever be so effectively designed that they cease to attract conscious attention in their final ubiquity? Certainly, the theory of design for ISs is still in its infancy; hence the need for the present volume.

Before considering their design, we first need to make clear what we mean by ISs. Many systems are interactive but outside the remit of this book. Motor cars, power drills, electric kettles, and so on are all interactive systems that will not be the subject of this chapter. By ISs we surely mean interactive digital systems (IDSs) that make use of digital representations and operations on these in order to effectively perform their allotted tasks. IDSs will therefore identify everything from ATMs and remote controlled TV teletext systems to PC and game console applications to onboard computers in cars and fly-by-wire aircraft.

An interesting subset of IDSs are interactive digital environments (IDEs) by which we mean an IDS that creates a large-scale digital environment that takes time and effort to explore and otherwise interact with. Examples of IDEs are videogames and virtual environments (VEs) in general, computer-based learning applications, and large-scale sites on the World Wide Web. These are interesting because the scale and complexity of their content demands that their effective design transcend established user interface techniques. Indeed, for VEs the very term *design* is a problem because it has to be interpreted in two quite distinct ways. First of all there is the notion of designing something to create the desired perceptual and aesthetic responses: essential for computer games. Secondly, there is the engineering notion of design as the creation of plans and models from which to test and build the desired artefact and ensure its correct functioning. Both forms of design are of equal importance to the design of effective VEs. It is the tension between these two notions of design and the resolution of this 'design tension' that is the central problem addressed in this chapter.

The need to resolve or at least alleviate this tension leads to a consideration of methods for VE design. It is assumed by some that the design of effective VEs will necessitate a development methodology akin to those used (or not) by software engineers. This is not necessarily the case. A craft-based approach based on the application of good practice—perhaps acquired through some form of apprenticeship—might do equally well. The computer games industry seems to prosper on just such an approach. The approach taken in this chapter is that an appropriate form of development methodology for VEs is viable, but that that methodology needs to accommodate—and certainly not stifle—the creative flair that is at the heart of aesthetic design of such large and complex systems.

This chapter therefore concerns itself with the investigation of what form an appropriate design methodology for VEs would take and the obstacles to establishing such a methodology. It is thus primarily concerned with a methodology of design—in other words, the meta-study of VE design methods rather than the outline of a particular method, although this is an obvious objective.

This chapter first undertakes an overview of the meaning of the various terms involved in the discussion: method, methodology, model, and language, among others. It then goes on to discuss the particular form an 'underlying model' for a VE method would have to take. Following this the issue of the form a language for expressing VE design decisions might take with regard to the underlying model put forward in the third section is addressed. The chapter then goes on to establish a process model for VE design and the 'practice of methodology' it to a large extent determines. It finally attempts to address future trends in the field and is followed by a short conclusion to the issues raised.

TERMINOLOGY

A methodology of design for VEs concerns itself with the study of methods for the design of VEs; in other words, the nature, definition, and application of such methods. This notion of methodology, while being quite correct, is at variance with a related but somewhat different notion that commonly views a methodology as a configurable method. In this chapter we use the approach of the former in order reach some conclusions with respect to achieving the latter.

If we are considering the study of methods for VE design, what do we mean by method in the first place? In software engineering the concepts of method and model are commonly understood, although the formality with which they are defined and applied varies considerably.

With respect to the question posed above, we will adopt the definition of Kronlof (1993) who defined a method as consisting of the following:

- An underlying model
- A language
- A process model
- Heuristics

Fencott et al. (1994) discuss these terms in the context of investigating the integration of structured and formal methods for software engineering. Methods integration will also be at the heart of the investigations of this chapter. Before using this characterization of method to address VE design, we will discuss the concept of model in some detail as it appears twice above in seemingly different contexts.

Models have been at the heart of much of human understanding and enquiry from very ancient times. Cultures very often attempt to explain the world and human beings' place in it by means of complex mythologies. Such mythologies are essentially abstractions—etiological fables (Carruthers, 1998)—that allow complex and inexplicable phenomena to be understood in terms of a more accessible set of characters and stories set around them. Very often the underlying explanation of phenomena will map onto supernatural beings and phenomena which thus replace unfathomable cause with commonly held narrative.

With time, more rigorous forms of modelling were invented. The ancient Mesopotamians developed sophisticated mathematics as a technique for modelling trade involving large numbers of items and customers (Davis & Hersh, 1983). This early theory of mathematics was thus being used to build abstract models of trade and stock control. The ancient Greeks and following them the Arabic world continued to develop models—mathematical and otherwise—for a variety of phenomena ranging from cosmology to music and poetry. Meter and rhyming schemes for poetry, for example, are models that facilitate the construction of new poems within established forms. This leads us naturally to ask what we mean by the term model, and how and why models are so generally useful?

The Concise Oxford dictionary variously describes a model as "a representation of structure"; "a summary, epitome, or abstract"; and "something that accurately resembles something else." Formal logic uses the term *model* to mean the system of rules by which meaning is mapped onto the syntactic constructions expressed within a particular logic. It is thus possible for a model to be highly formal—that is, expressed in mathematics—or highly informal, but not presumably both. Scientific models may be more pragmatic in that they are related to some aspect of reality by means of observational data, which in turn causes the hypothesis upon which the model is constructed to be reformulated and so on. In other words they are empirical rather than strictly formal and thus sit somewhere between the extremes of the formal-informal axis.

As already mooted with respect to etiological fables, models may be quite instrumental in the sense that the application of the model as

an analysis technique—and the results obtained therein—may be more important than the degree to which the model accurately reflects reality; psychoanalysis is an obvious example. Semiotics (Chandler, 2002) is perhaps another case in point because it has never been ascertained whether or not signs as defined by semioticians actually represent structures or functions within the human brain. There is some evidence to support this (e.g., Damasio, 1994). Nonetheless, semiotic analysis of communications artefacts—texts to semioticians—is a very valuable and general technique for gaining insights into the way in which humans communicate and make meaning using a whole range of media. Semiotics is very important to this chapter.

With respect to Kronlof's characterization of method, we can see that the term model is used in two rather different ways:

1. An 'underlying model' is a semantic structure to which terms of the language of the method are mapped in order to assign meaning to them.
2. A 'process model' is an abstract representation of the activities undertaken as part of the model along an expression of their ordering.

The first use of the term model given above is a formal notion, while the second is the more intuitive notion of an abstraction of some more complex system, both discussed in our aside above. If we were to take the language and its underlying model together, we would arrive at the second form of model which is essentially a notation for simplifying and elucidating a more complex system. But what language and underlying model are we to use for VE design? The role of the former is to facilitate the creation and expression of design decisions. The role of the latter is less obvious, but its nature has a direct bearing on the applicability of the method in general. The two parts of this question are addressed in the succeeding sections of this chapter.

The process model of a method is most often expressed as a simple diagram, a graph where the nodes name particular activities and the arcs indicate the relative ordering over time of these activities. The graph is thus a focused simplification of a complex set of activities and the relationships between them and their products. What process model might be suitable for VEs? Kaur (1998) put forward a tentative process model for VEs as an ordered list of activities. These activities and their ordering were deduced from questionnaire data drawn from a limited number of VE developers. Fencott (1999b) put forward a process model that was more representative of the design tensions inherent to VEs. We will return to the process model after the sections devoted to language and underlying model.

Heuristics are essentially advice and guidelines on the successful application of the model to real problems. In terms of VE design, we can observe that there are a lot of such heuristics around in terms of standalone advice that is almost invariably devoid of a methodological context with respect to VEs. There are exceptions to this, the 'SENDA' method of Sanchez-Segura et al. (2003; also, see Chapter 4) for example.

The 'design tension' identified above as the driving force in the methodology of VE design has its antecedents. In the early 1990s there was a debate as to whether formal methods or structured methods for software design were most appropriate. The former use logic and set theory to build mathematical models of software systems, while the latter use diagrams, pseudo code, and other 'non-formal' notations to the similar ends. Integrated methods research attempted to combine these approaches to maximize the strengths and minimize the weaknesses of both (Fencott et al., 1992, 1994). In this chapter we draw on the experiences gained in the earlier research in order to address the design tension directly.

In this section we have posed a number of questions with respect to a possible VE design method:

1. What language and underlying model are we to use for VE design?
2. What process model is appropriate for VE design?
3. What sort of heuristics do we need and are any of those extant adaptable to the model we hypothesize in 1 and 2 above?

In this chapter we specifically deal with Questions 1 and 2. Question 3 will be for future consideration, as it depends on the answers to Questions 1 and 2.

THE UNDERLYING MODEL

The question of what an underlying model might be for a VE design methodology might seem of purely theoretical interest, but attempting to answer it necessitates a consideration of the design tension highlighted in the previous two sections. We have to find an underlying model that expresses the meaning of a VE design in terms of both:

- **Engineering:** As a computer system composed of program and hardware, understood largely by those trained in computer science and related disciplines;
- **Aesthetics:** As an interactive communications medium, understood by those trained in the creative arts.

We appear to have confounded the issue, as we now seem to need an underlying model that not only addresses two different design issues, but that is understood differently by two quite different groups of professionals. Is one underlying model possible, and who on earth is going to understand it? In fact there have been various attempts to reconcile the two with varying degrees of success, but it's useful for our purposes to consider them separately for the time being.

We can begin to suggest possible underlying models, bearing in the mind the tension already identified. VEs and IDSs in general have interaction machines (IMs) as their underlying model (Goldin et al., 2001) in terms of computational functionality, but we also need a model that operates at the perceptual, meaning-making level. Semiotics (Chandler, 2002) is highly appropriate for the latter. Interaction Machines encompass a set of possible computational systems—more expressive than Turing Machines—that allow for the persistence of state and unlimited user inputs that characterize interactive media, IDSs in general and VEs in particular. Semiotics is the study of sign systems and the way humans find meaning in them. The two might not be so incompatible as a cursory glance might seem to suggest. We will briefly consider each separately and then consider their integration.

For much of the latter half of the twentieth century, it was the received wisdom that Turing Machines captured the notion and limits of what is computable. In the 1990s a number of researchers began to develop models which showed that Turing Machines were not expressive enough to model interactive computer systems. In fact it was shown that the simplest interactive program:

```
P := input(x:Boolean); output(x);
P
```

which recursively inputs a Boolean value for x and simply outputs that same value, cannot be programmed using any Turing Machine. That this is so even for a very simple datatype such as Boolean might be somewhat surprising. The reason is that although each input and output is finite—a requirement for conventional Turing Machine input—there might be an infinite number of them, and it is impossible to represent such an infinite set of choices on a sequential, yet infinite tape.

Goldin et al. (2001) have shown that Turing Machines can be extended to model interaction by defining Persistent Turing Machines (PTMs), which employ dynamic streams to model inputs

and outputs, and a tape to remember the current state ready for the commencement of a new computation. PTMs are an example of the general class of IMs.

PTMs are certainly not the only possible characterization of IMs. We could, for instance, have used an approach based on concurrent systems in the manner of Milner (1989). In many respects this would be better as it not only captures the notion of VEs as IMs, but also allows us to consider them as being the composition of a number of embedded systems—autonomous agents and non-playable characters, for instance. PTMs are, however, better suited as a brief illustration of the concept for our present purposes.

Human beings ceaselessly work to find meaning in any situation they might find themselves, in any communications media they might find themselves using, and in even mundane situations such as walking down the street or sitting on a train or bus. Semiotics is the study of this meaning-making process, and signs are the basic unit of the theory (e.g., Eco, 1977; Barthes, 1987). The most common characterization of signs consists of two components, a:

- **Signifier:** That which we can perceive in the world around us using any of our senses;
- **Signified:** The meaning(s) we form in our minds as a response to perceiving the signified.

Communications artefacts, texts to semioticians, are made up of signs and can be anything we humans find meaningful, for instance: novels, films, body language and facial expressions, and VEs.

Semiotics provides us with a means of understanding the output of a VE, the digital displays, and the signs of intervention, as we shall call them, that the user generates by means of the input technology. VEs are a particular form of IM that attempt to restrict its users' environments to the digital displays it generates in response to user input. We thus have a partially closed system. Semiotics can provide a means of analysing how a user might make meaning out of such a system and thus make meaningful choices about how to interact with it. We can thus refer to our underlying model as a Semiotically Closed Interaction Machine (SCIM).

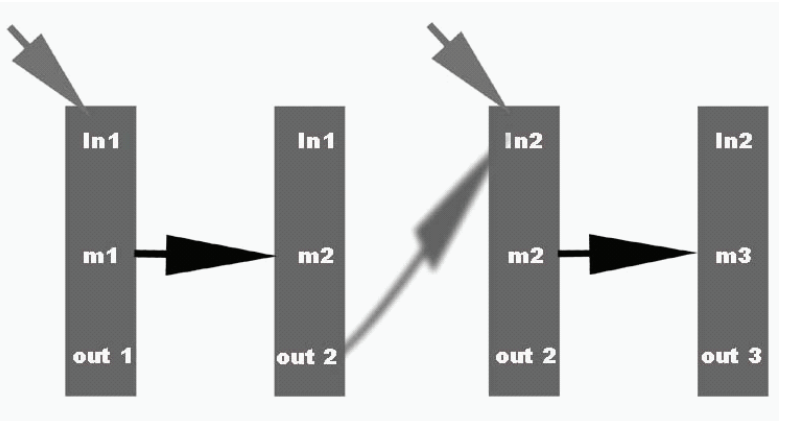
Figure 1 shows the relationship between semiotics and IMs. The two downward pointing arrows represent inputs by the user, in1 and in2. The horizontal, black arrows represent computation steps that result in the generation of new outputs, out2 and out3. The fuzzy, curved arrow represents the semiotic closure between out2 and in2; in other words the cognitive process of finding meanings in out2 and formulating a response to them as in2. On the one hand we have the human, meaning-making process and on the other the non-semiotic act of using the signs of intervention to create a new input to the IM and thus instigate a further macro-computation step. Note that the diagram is a simplification, as in VEs in general outputs may also be produced without direct input from the user.

In SCIMs such as VEs, the semiotic link is very strong, whereas in IDSs in general, the link may be far weaker and intermittent. There is no semiotic link between individual customer transactions at an ATM, for instance. There is also no recognizable semiotic link between a customer inserting his or her debit card, the PIN input, and the amount of money requested; ATMs are not SCIMs.

Both IMs and semiotics are appropriate as a choice of an integrated, underlying model because they do not constrain us to particular programming languages or computational platforms on the one hand, nor particular modes of communication on the other. That will be the business of the next section when we consider the nature of a language suitable for expressing VE design decisions.

An integrated underlying model is not the only approach. There is a field of enquiry called computational semiotics that has as one of its

Figure 1. Semiotically closed interaction machines



concerns the integration of semiotics and computer science; this can operate at the level of the underlying model or at the level of language within a methodological context while sometimes at both. For instance, Goguen (1999) defines 'algebraic semiotics' as semiotics formalized using the algebraic specification language OBJ. He has outlined the application of this formalism to user interface design and VE design. As another example, Doben-Henisch (1999) has attempted to integrate semiotics with Turing Machines. The problem with the latter is that Turing Machines are not expressive enough to model VEs. The problem with the former as an underlying model for VEs is that the formalism makes use of difficult mathematical concepts, such as category theory, which obscure the insights into the nature of VE that our integrated approach highlights—difficult, that is, for those VE designers without a strong mathematical background.

The integrated underlying model we have adopted is a very practical one, as it preserves the 'design tension' rather than allowing the engineering or the aesthetic dimension to dominate.

THE LANGUAGE

We move now to the nature of languages for expressing VE design decisions. A review of existing work on VE design (Fencott, 2003b) reveals that while there is a quantity of research and commentary on the human factors affecting design, for instance, there is very little that is directly relevant to VE content modelling, which is at the heart of this chapter. There are examples of the construction and application of methods or guidelines for realizing certain aspects of VE design; some of these are:

1. Various work on usability for VEs (e.g., Workshop on Usability Evaluation of Virtual Environments, 1998)
2. Structured methods for VEs (e.g., Workshop on Structured Design of Virtual Environments, 2001)
3. Various commentaries from the computer games world (e.g., Gammasutra, Rollings, & Adams, 2003)
4. Semiotics of games and new media (e.g., Lindley et al., 2001)

In light of the discussion in the previous section, we can make the following observations: 1 and 2 are insufficient to express VE design decisions because they do not address aesthetics adequately; 3 provides some very useful insights; 4 gives us a way to alleviate the inadequacies of 1 and 2.

If we continue with the integrated approach adopted for the underlying model in the previous section, we need a language to express the programming (the engineering) side of a VE and one to express its aesthetic dimension. The standard for the former should, most likely, be some form of object-oriented programming language and the standard methodology for such languages is the Unified Modelling Language (UML). In fact, Goldin, Keil, and Wegner (2001) document the suitability of UML as a language for expressing designs that have IMs as their underlying model. UML would seem a good choice of language for this aspect of VE design.

Aesthetics of VEs has been a constant theme of this chapter, and we now discuss them in some detail. Church calls for a set of "formal, abstract, design tools" (FADTs) that will not only guide the design of successful games, but which will also enable designers to compare and contrast computer games from diverse genres (Church, 1999). Church's FADTs are perhaps better understood as an aesthetic characterization of computer games and are:

- **Intention:** Being able to establish goals and plan their achievement;
- **Perceivable consequence:** A clear reaction from the game world to the action of the player;
- **Story:** The narrative thread, both designer-driven and user-driven, that binds events together.

Other computer games designers talk in a similar vein: of players needing to feel in control, of maintaining the emotional feel of a game and/or level, of providing suitable and timely rewards for effort, and of a perceivable gross structure that allows players to identify what is required of them at the beginning of a level, plan to achieve this, and understand the significance of their achievement (Saltzman, 1999). Intentions and perceivable consequences are the building blocks for this.

Brenda Laurel introduced the term 'narrative potential' to capture the idea that VEs can offer users the possibility of building their own stories out of virtual experiences (Laurel, 1992). We will adopt narrative potential rather than 'story' as part of the aesthetics of VEs.

From the field of media studies, Murray (1996) identifies the following aesthetic characterization of interactive media as:

- **Immersion:** The feeling of being completely absorbed (almost literally immersed) in the content (we will use the term presence for reasons detailed below);
- **Agency:** Being able to affect change in the VE;
- **Transformation:** Being able to become someone or something else.

Lombard and Ditton (1997) define presence as *the perceptual illusion of non-mediation*. This characterizes presence as the state of mind of a visitor to a VE as not noticing or choosing not to notice that that which they are experiencing and interacting with is artificially generated. They document the evaluation of the embodying interface of a VE in terms of presence seen largely as the degree of fidelity of sensory immersion. Much of the research to date into presence is particularly concerned with the embodying interface as well as researches into the mental state of people who are present in VEs. Immersion is thus the degree to which the technology of the embodying interface mediates the stimuli to the senses. Slater has shown that high degrees of sensory immersion heighten the emotional involvement with a VE (Slater et al., 1999).

However, as presence is a mental state, it is therefore a direct result of perception rather than sensation. In other words, the mental constructions that people build from stimuli are more important than the stimuli themselves. It is the patterns that we, as VE constructors, build into the various cues that make up the available sensory bandwidth for

a given VE that help or hinder perception and thus presence. These patterns are the result of what is built into the VE and the way the user behaves in response to them. The fidelity of the sensory input is obviously a contributing factor, but by no means the most important. In the context of the working VE builder, being able to identify and make effective use of the causes of presence is more important than the nature of presence itself. This means that it is the effective consideration of the perceptual consequences of what we build into VEs that will give rise to the sense of presence that we are looking for. In this sense it is the content of VEs that has the greatest effect on the generation of presence. Thus, for our purposes, content is the object of perception.

Agency is the fundamental aesthetic pleasure of VEs and IDSs in general and the one from which all the others derive. Agency actually equates quite nicely to Church's intention and perceivable consequence; agency is in part the interplay between intention and perceivable consequence.

Transformation is important to many communications media. One of the great pleasures of novels is seeing the world through someone else's eyes, to view the world through the eyes of another creature, machine, or alien being. VEs in particular are ideally suited to this, and much of the success of 3D computer games is due to the player being able to be the hero or villain in some great and dangerous adventure. In such games the player cannot only play an alien, but through the real-time graphics actually see the world as the alien would see it. It seems certain, for instance, that one of the reasons for the success of the classic Hubble Space Telescope Virtual Training Environment (Loftin et al., 1994) was that members of the ground-based flight team could actually become astronauts for a while, and experience some of the drama and spectacle of a space walk. To the author's knowledge and despite the insightful research into the effect and effectiveness of the Hubble, the question "Did you enjoy being an astronaut for a change?" was never asked. Yet it seems highly likely that this was a major experience for the subjects.

Finally, in this brief review of aesthetics for IDSs, we must include Turkle's (1995) observation that being present with others—sentient beings, robots, creatures, and autonomous agents in general—is something that has drawn users to IDSs since the earliest days of Eliza and MUDs.

Bringing these various aesthetic viewpoints together, we can characterize the aesthetics of VEs as:

- **Agency:** Which itself consists of:
  - **Intention:** Being able to set goals and work towards their attainment.
  - **Perceivable consequence:** Being rewarded for one's mental and virtual activity by sensing the VE change appropriately as a result of the actions taken.
- **Narrative potential:** The sense that the VE is rich enough and consistent enough to facilitate purposive experience that will allow the user to construct her own narrative accounts of it.
- **Co-presence:** Being present with others.
- **Transformation:** Temporarily becoming someone or something else as a result of interacting with the VE.
- **Presence:** The perceptual illusion of non-mediation (Lombard & Ditton, 1997).

In terms of underlying theory, aesthetics are signifieds of a particular type; they are connotations that arise from interacting with VEs. Connotations, in semiotic theory, are deeper levels of meaning that humans build up from the level of denotation: the commonplace or everyday meanings of things.

On a more concrete level, Murray (1996) equates the structure of interactive media with the notion of the labyrinth and asserts that this structure works best when its complexity is somewhere between the 'single path maze' and

the 'rhizome' or entangled Web. Aarseth (1999) has proposed the notion of cybertext to capture the class of texts, not just digital, which require the visitor to work to establish their own path(s) through the possibilities offered. He calls this class of text ergodic from the Greek words meaning work and path. So we have a notion of a labyrinth that requires effort to explore. Equating the structure of VEs in general with the notion of a labyrinth of effort would seem useful, but poses several questions. First of all, what are the actual components with which VE designers build such experiential labyrinthine structure? Second, how do VE designers structure a VE so that the visitor follows an appropriate path and, moreover, accumulates an appropriate set of experiences so as to discover and remember the intended purpose of the VE?

Fencott (1999a, 2003a, 2003b) draws on these various aesthetic views to define a model of VE content, Perceptual Opportunities (POs), which focuses on the aesthetic design of the perceptual experiences over time which users are intended to accumulate.

Figure 2 characterizes the breakdown of POs in terms of:

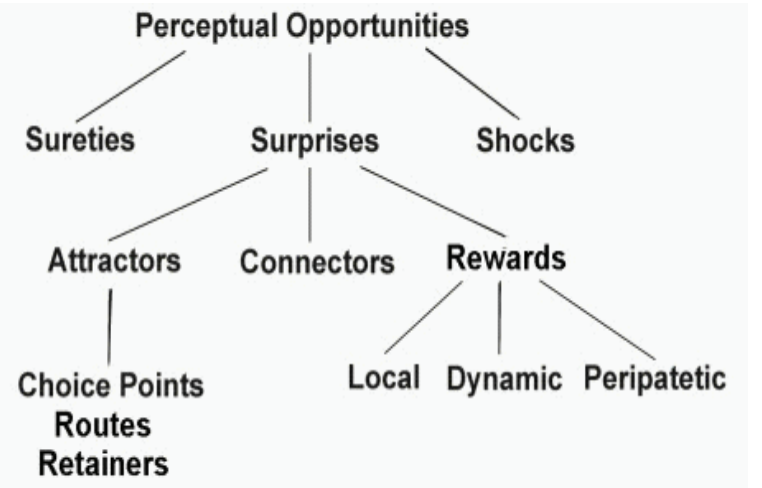
- **Sureties:** Designed to deliver belief in a VE, equated with unconscious experience (e.g., Spinney, 1998; Blackmore, 1999)
- **Surprises:** Designed to deliver the essential purpose of the VE
- **Shocks:** Perceptual bugs that undermine the first two

Surprises are further broken down into:

- **Attractors:** Literally content that attracts attention
- **Connectors:** Content that supports the achievement of goals
- **Rewards:** Content that literally rewards users for effort

Attractors can be characterized in two ways: By the way they attract attention—they might be mysterious, awesome, active, alien, complex—collections of attractors—and so on. They can also be characterized by the basic emotions they stimulate, typically fear and desire. Rewards can be information, access to new areas of the VE, new activities enabled, and so on. Connectors can be as simple as railings, footpaths, and street signs, but can also be dynamic maps, indicators of

Figure 2. Perceptual opportunities



health, wealth, and so on. Attractors are the means by which users are led to form intentions. The perceivable consequences of a player attempting to realize an intention leads to the identification of rewards which leads to the identification of new attractors and so on. Thus agency and POs are very strongly associated.

POs can be organized into higher level structures, perceptual maps, which characterize patterns of behaviour that users exhibit when interacting with a VE. A perceptual map can be made up of:

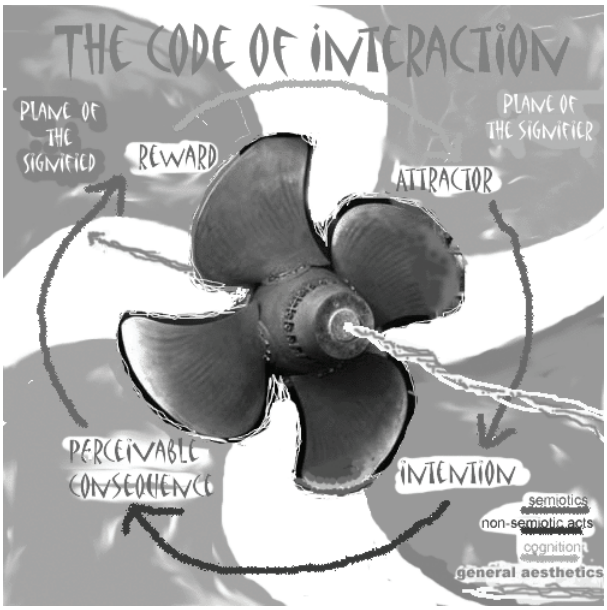
- **Choice points:** Basically the choice between intentions stimulated by one or more attractors.
- **Challenge points:** Intentions that have to be satisfied.
- **Routes:** Linear sequences of attractors.
- **Retainers:** Mini-missions or mini-games, tightly grouped attractor-reward pairs, puzzles, and so forth.

The arrangement of such structures are thus a realization of Murray's rhizome and lead to the other aesthetic pleasures of narrative potential, co-presence, transformation, and presence.

In a later publication, the same author asserts that POs, as well as the aesthetics identified above, have semiotics as their underlying model (Fencott, 2003b). Essentially, POs and in particular surprises are connotations that humans derive through interacting with VEs. POs interface very closely with the aesthetic pleasure of agency, but at a more abstract level of VE content. Figure 3 illustrates the relationships between POs, aesthetics, and semiotics at the level of the language of a VE design method:

- The two arrows linking attractor and intention and perceivable consequence and reward are semiotic acts, meaning making, on the part of people interacting with a VE. Attractors and perceivable consequences are signifiers, while intentions and rewards are signifieds.

Figure 3. The code of interaction



- The arrow linking reward and attractor indicates cognition, though of course cognition is a continuous process and not a segment of a cycle as this diagram would seem to suggest.
- The arrow linking intention and perceivable consequence represents what Tronstad (2001) calls non-semiotic acts that are essentially the site of the IM, the computer-based system in the wider IDE. The term non-semiotic is used because, while the user might draw some significations from pressing interface buttons and so on, the computer responds algorithmically.
- The arrow that runs through the cyclic plane of the above relationships from right to left represents the development over time of the other aesthetic properties of narrative potential, co-presence, transformation, and presence.

On the level of aesthetics and POs, we see the following. Having formed an intention, a user will provide input to the VE, which will trigger the execution of one or more calculations, non-semiotic acts on the part of the computer. This will result in a change (perceivable consequence) in the various digital elements of the VE's display which provide signifiers (rewards) to start off the whole semiotic and cognitive process once again through the identification of attractors.

Figure 3 shows quite clearly the dependant relationship between semiotic and non-semiotic acts, which Tronstad (2001) sees as being fundamental to interactive digital experience. If we compare Figure 3 with Figure 1, we see that what has changed is that the arrow that represented the semiotic closure of the output and input step in the latter has been dramatically expanded in the former; it is almost as if it has been turned 'inside out'. Figure 3 characterizes the 'code of interaction'. In semiotics, codes are the often innate rules that allow us to make meaning of signifiers. Interaction is a complex process and the diagram reflects this. So much so, in fact, that Fencott (2004) devotes a whole chapter to the 'code of interaction. In the context of our present discussions, the various components and the relationships between them that make up the code constitute the general aesthetic side of the language of our method.

Semiotics not only provide an underlying model for POs and aesthetics, they also operate at the level of the language of a method as well. In interacting with VEs we not only recognize the code of interaction—connotations specific to VEs and IDSs in general—but we also find meanings that correspond to world of the 'real' outside the world of the VE. We recognize shops and cars and people and furniture and so on and so on. It is semiotics itself that is used as 'language' in this type of meaning-making.

Therefore, in addressing the second question concerning the nature of the language of a VE design method, we now need to consider how POs, aesthetics, and semiotics on the one hand and UML on the other might work together. In this respect, the central issue that needs to be addressed concerns what we might call the 'object problem'. Objects or rather object-oriented design (OOD) might seem a very promising candidate for our language for representing VEs at the design phase. OOD applies at all stages in the VE production lifecycle, addresses both coding and user-cantered issues, and has been applied directly to VE design and implementation (McIntosh).

However, in the act of perception, people do not break the world down into nicely programmable units. They group things together into perceivable units, complex attractors, which focus their attention. A crowd of autonomous agents—non-playable characters (NPCs) in computer games parlance—are perceivable as a single entity, but are unlikely to be a single object in an OO model. Certainly a crowd of NPCs in a busy shopping centre with all its shop fronts, street furniture, paving, and so on is not going to be an object in an OO specification for a shopping centre. However,

each of the entities that makes up the perceivable unit that is the crowded shopping centre will have to be identified in terms of its capacity (or not) for interaction as a basis for its incorporation into a functioning scene graph.

On the one hand, we have the Unified Modelling Language (UML), which models structural, engineering aspects of a scene graph, and on the other hand, we have POs and so on which model content at the level of perception, of aesthetics (Fencott, 1999b, 2003b). There is in fact a bridge, a semiotic bridge, which links the two, and this is Andersen's Computer Based Signs (CBSs) (Andersen, 1997), which model interactive aspects of individual signs (objects) in IDSs in general and thus VEs. Fencott (2003a) discusses this relationship and its relevance to VE design.

Andersen arrives at the following classification of signs in IDSs:

- **Interactive:** Signs that can be controlled by the user and can affect other signs; such signs are subject to the signs of intervention.
- **Actor:** Signs that to a limited extent are autonomous and can affect other signs.
- **Controller:** Signs that constrain other signs but do not themselves change nor can they be affected by other signs.
- **Object:** Signs that can be affected but cannot affect others.
- **Ghost:** A sign that affects others, but only becomes apparent by its effects on others; a sign particular to IDSs. Essentially a controller that signifies its presence solely through effect.
- **Layout:** Non-interactive signs.

CBSs essentially constitute six distinct classes that will be used to instance all objects in a VE implementation. The integration of the three elements of our language of VE design can now be summarized thus:

- Each content item in the perceptual model is assigned to a CBS class.
- Other aesthetic attributes of content items carry over directly to UML, that is, colour, form, and so on.
- General information in the perceptual model carries over to UML to become the game engine, the visualiser.
- Other such information carries over directly to UML in terms of the semiotic realization of the VE: mood, myths, and hyperrealities.

So the language of our VE design method is an amalgam of OO and POs and so on—with some bridging by CBSs. It is, in fact, an integrated method, a process, rather than a statically characterisable relationship. In this way we have carried the design tension identified early in this chapter, and clarified through to the language stage. It seems that we might be able to make this tension work for us rather than it being a hindrance to try to do away with.

THE PROCESS MODEL

The process model captures the relationship over time between the constituent activities of a method. In a sense it captures the essence of the 'practice of methodology', the choosing of how to apply a method. As part of a study of VE design practice, Kaur (1998) constructs the following outline VE design methodology:

1. Requirements specification;
2. Gathering of reference material from real-world objects;
3. Structuring the graphical model and, sometimes, dividing it between designers;
4. Building objects and positioning them in the VE;
5. Enhancing the environment with texture, lighting, sound, and interaction, and optimising the environment.

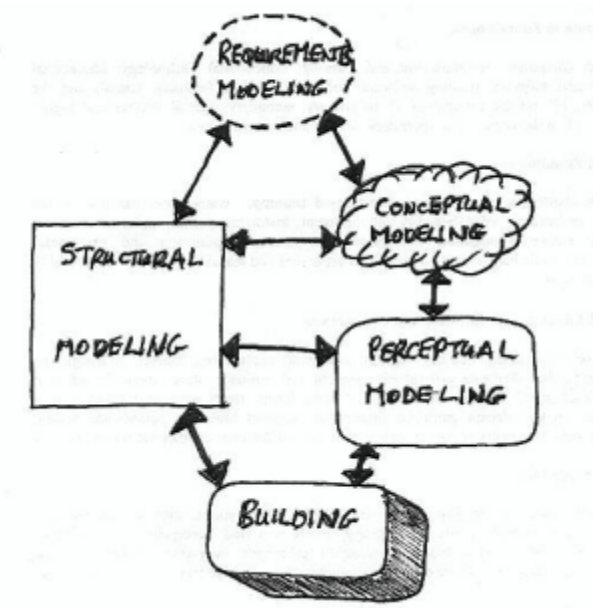
She also notes that there might be a narrative design component missing here, but this is probably because of the small scale of the VEs in the study. Certainly the narrative aspects of 3D game design are considered as soon as the principle subject and genre are established. Computer games are almost certainly the major examples of VEs large enough to benefit from software engineering practice.

With these arguments in mind, Fencott (1999b) offered a prototype design methodology for VEs which attempts to resolve the two-sided design problem for IDEs by juxtaposing structural and perceptual modelling, and attempting to empathize with current practice. The methodology is also based on practical experience gained in building a variety of desktop VEs, and in particular a virtual tourism project, as well as teaching VE design to several hundred undergraduate and master's students over a number of years. Figure 4 characterizes this suggested process model, and we now go on to revisit the original, tentative discussions that were offered in the 1999 paper, in the light of the discussions concerning the possible underlying model and language required for a design method for VEs laid out above.

In terms of the design tension, the route down the left-hand side of the diagram represents engineering design and the route down the right-hand side represents aesthetic design. The horizontal arrows represent interactions that seek to resolve the tension.

- **Requirements modelling** equates to Point 1 in Kaur's methodology above and parallels very closely the software engineering concept. One of the chief requirements is that purpose should be clearly established here. In terms of our integrated underlying model, we might here conduct as 'use case analysis' in UML and commence the analysis of our intended VE in terms of Barthe's notion of *myth*—connotations so seemingly natural as to be unquestioned (Barthe, 1987)—and perhaps Baudrillard's notion of hyperreality (Baudrillard, 1995). Both are concerned with the cultural basis upon which a VE's belief system will be grounded. At this stage we

Figure 4. A process model for VE design



are thus making direct use of the underlying model and the techniques associated with it. The semiotic and software engineering viewpoints are left unresolved until the latter stages of structural and perceptual modelling.

- **Conceptual modelling** equates to Point 2 in Kaur's methodology and is effectively the background research activity common to many design projects, but in particular those with an aesthetic component. It is the gathering of materials, taking of photographs, sketches, sound and video recordings, and so forth. It might also include the construction of mood boards as well as potential storyboards. This is where the VE builder or builders get to know the world they have to build. Note that the world to be built might have no real-world counterpart, which will of course impact on the kinds of activities that might be undertaken here. The artists' accounts and the techniques employed by animators are sources of applicable techniques (e.g., Moser, 1996). An important outcome of this stage will be a choice of genre, to best achieve the purpose established at the requirements stage, with which to inform the nature of the meta-narrative structure to be developed in the perceptual modelling phase.

  The end of this phase is effectively concerned with the semiotic activity of translating the decisions concerning myth and/or hyperreality—from the requirements phase—into connotation, metaphor, and metonymy.
- **Perceptual modelling** is the act of building up a model of the nature of the perceptual opportunities and their inter-relationships. It equates very roughly to Point 5 in Kaur's methodology. It is of course modelling the intended users' experience of the VE. In Fencott (2003a) perceptual maps, for instance attractor graphs, are used to build up a meta-narrative structure of POs, analogous to the comprehensible labyrinth of Murray (1997), which are categorized according to the role they play in the planned scheme of possible user activity. Perceptual opportunities deal not only with conscious experience—derived from *the specifically designed infidelities* of Whitlock et al. (1996)—but also with unconscious experience, sureties, which deliver belief in the VE—perceptual realism in Lombard and Ditton (1997)—irrespective of any real-world counterpart. The existence and importance of unconscious experience is identified and modelled by considering sureties.
- **Structural modelling**, Point 3 in Kaur's methodology, covers a variety of activities that relate to the underlying realization of the VE that the delivery platform uses to construct the run-time sensory stimuli. Structural modelling would seem to commence alongside conceptual modelling and to run on alongside perceptual modelling. It starts with decisions on scale, the construction of plans, and diagrams. It draws on Andersen's CBSs to further decompose the perceptual map constructed in the perceptual modelling phase in terms of the way in which particular objects implement gross structure of attractors and rewards identified in perceptual modelling.

  The conclusion of the structural modelling phase will result in a scene graph diagram that lays out the code structure of the VE and its programmed behavioural components. In terms of software engineering practice, UML has already been identified as a candidate language here. In later stages, object models would lay out the actual structure of nodes in the scene graph as well as class diagrams for programmed components.
- **Building** here relates more closely to the software engineering coding phase that should occur after all requirements, specification, and design activities have been

completed. Building refers to authoring using a WIMP-based tool, direct coding of scene graph and program code itself, in VRML and Java/Javascript for example, and using an API such as World Tool Kit.

We will now consider some of the flows (arrows) in this process model, first of all the structural-conceptual flow. The conceptual modelling stage can deliver important high-level plans for the layout of the VE as well as the principle entities that will need to be present to reinforce the results of *use-case-analysis*, for instance. The structural-perceptual flow delivers object denotations to do with such attributes as appearance and sound. It will deliver object connotations concerned with the way objects contribute to the overall purpose of the VE. Importantly it will also—via CBSs—deliver attributes concerned with interactive capabilities of objects.

Finally, we note that we do not address the question of heuristics in this chapter. There are two reasons: first of all our method is too methodological at the moment to be able to be supported by practical advice; secondly there is a wealth of help and advice on VE design and games design in particular, and it will be necessary to investigate how it might integrate with the method under consideration.

FUTURE DESIGN METHODOLOGY

It is the author's suspicion that one of the foreseeable trends will be ever more sophisticated VE authoring tools, which will mean that the explicit use of OO techniques such as UML will be more and more hidden from the author. Much of the time the scene graph, whether at the level of an OO specification or at the level of OO coded implementation, will only be available to authors via specific views rather than as a coherent whole. More and more it will be the perceptual modelling and the interface between this and the structural views that will be made explicit and malleable. The process model discussed above shows the nature of this interface and provides clues as to how this might be supported by authoring technology.

However, in terms of authoring tools, there is a serious problem that we have not identified nor discussed so far. This is the problem of authoring agency, which lags far behind the authoring possibilities on offer for 3D modelling, texture mapping, shading, and rendering, to name but a few. Nothing approaching the sophisticated tools on offer for these exists for authoring agency. Typical examples of this are easy to find in a wide range of VE authoring tools for both games and VR. In the excellent Unreal Editor for example, the only agencies we can easily implement are such concerned with opening doors, travelling in lifts (elevators), and shooting guns. Unreal is a *first-person-shooter* and it has in-built agencies typical of its genre. If an author wants to implement additional agency, then she has to program it in Unrealscript, a Java variant.

Yet a theoretical analysis of games genres has shown that agency is exactly what characterizes games (Fencott, 2004). Any game design method should not only incorporate the analysis and design of appropriate agency in its process model, but should encourage authors to reconsider it throughout the lifecycle from early requirements analysis through to later modelling stages. By focusing on agency in terms of the aesthetic pleasures of intention and perceivable consequence, and in terms of the POs of attractors and rewards, the process model does indeed ask the designer to consider agency in a fundamental way that authoring tools do not at present support.

In terms of underlying theory, two significant trends can be identified. The growing interest in the investigation, formalization, and application of interaction machine theory (e.g., Goldin et al., 2001) and the emergence of semiotics and computational semiotics as a tool to analyse and design VEs and IDSs in general—for example the COSIGN series of conferences (COSIGN).

Of particular interest will be the further investigation of the possible integration of interaction machines and semiotics, which in effect amounts to the nature of the interplay between empirical computer science and interactive media aesthetics. As has already been pointed out, the tempting approach is to formalize semiotics as computation (e.g., Dogen-Henisch, 1999; Goguen, 1999), but this does not capture or investigate the playfully surprising relationship people have with IDSs and IDEs in particular.

CONCLUSION

A little tension can be a good thing; too much can be very destructive. We need to keep the VE design tension apparent throughout the analysis, conceptual, and perceptual design stages to be more or less resolved in the structural modelling stage. It must not be allowed to tear the process apart. On the other hand an imbalance biasing one pole of the tension or the other will result in an equally unbalanced VE—either well engineered and boring, or fascinating but badly made. The author believes that the design tension will manifest itself in a benign way in a well-designed VE and that users will recognize and appreciate that manifestation.

In effect VE design methodology is encouraging us to confront and meld a great rift in contemporary Western culture, namely that between the arts and the sciences. Of course, at present it is inviting us to do this in terms of two particular forms of abstraction which represent the two sides of the divide. That we should confront reality through virtual reality might come as a surprise, but the concept has been around since the early days of virtual environments and was clearly articulated by Lauria (1997) when she envisioned virtual reality as a 'metaphysical testbed'.

On a less grandiose scale, it may well be that no design method for VEs ever becomes a real practicality or if it does is ever widely adopted by the developer community. Surely, however, the investigation of the methodology of VE design will inform us far better than we are now as to the fundamental nature of VEs and thus be of benefit to us when we come to design future interactive systems.

REFERENCES

Aarseth, E.J. (1997). *Cybertext: Perspectives on Ergodic literature*. Baltimore, MD: John Hopkins University Press.

Andersen, P.B. (1997). *A theory of computer semiotics.* Cambridge University Press.

Barthes, R. (1987). *Mythologies*. New York: Hill and Wang.

Baudrillard, J. (1995). The Gulf War did not take place (trans Patton P.). Indiana University Press.

Blackmore, S. (1999). *The meme machine.* Oxford University Press.

Carruthers, M. (1998). *The craft of thought: Meditation, rhetoric, and the making of images, 400-1200.* Cambridge University Press.

Chandler, D. (2002). *Semiotics*: *The basics*. Routledge.

Church, D. (1999). Formal abstract design tools. *Games Developer Magazine,* (August).

COSIGN. Retrieved from www.cosignconference.org

Damasio, A.R. (1994). *Descarte's error: Emotion, reason and the human brain.* Papermac.

Davis, P.J., & Hersh, R. (1983). *The mathematical experience.* Pelican Books.

Doben-Henisch, G. (1999). Alan Mathew Turing, the Turing Machine, and the concept of sign. Retrieved from www.inm.de/kip/SEMIOTIC/

DRESDEN_ FEBR99/CS_Turing_and_Sign_ febr99.html

Eco, U. (1977). *A theory of semiotics.* Macmillan Press.

Fencott, C. (1999a). Content and creativity in virtual environment design. *Proceedings of Virtual Systems and Multimedia '99,* University of Abertay Dundee, Scotland.

Fencott, C. (1999b). Towards a design methodology for virtual environments. *Proceedings of the International Workshop on User Friendly Design of Virtual Environments,* York, UK.

Fencott, C. (2003a). Virtual saltburn by the sea: Creative content design for virtual environments. *Creating and using virtual reality: A guide for the arts and humanities*. Oxbow Books, Arts and Humanities Data Service.

Fencott, C. (2003b). *Perceptual opportunities: A content model for the analysis and design of virtual environments.* PhD thesis, University of Teesside, UK.

Fencott, C. (2004). *Game invaders: Computer game theories.* In preparation.

Fencott, P.C., Fleming, C., & Gerrard, C. (1992). Practical formal methods for process control engineers. *Proceedings of SAFECOMP '92*, Zurich, Switzerland, October. City: Pergamon Press.

Fencott, P.C., Galloway, A.J., Lockyer, M.A., O'Brien, S.J., & Pearson, S. (1994). Formalizing the semantics of Ward/Mellor SA/RT essential model using a process algebra. Proceedings of Formal Methods Europe '94. *Lecture Notes in Computer Science, 873.* Berlin: Springer-Verlag.

Gammasutra. Retrieved from www.gamasutra.com

Goguen, J. (1999). An introduction to algebraic semiotics, with application to user interface design. Computation for metaphor, analogy and agents. *Springer Lecture Notes in Artificial Intelligence, 1562,* 242-291.

Goldin, D., Keil, D., & Wegner, P. (2001). An interactive viewpoint on the role of UML. *Unified Modelling Language: Systems analysis, design, and development issues.* Hershey, PA: Idea Group Publishing.

Goldin, D.Q., Smolka, S.A., Attie, P.C., & Wegner, P. (2001). Turing Machines, transition systems, and interaction. *Nordic Journal of Computing.*

Kaur, K. (1998). *Designing virtual environments for usability.* PhD Thesis, City University, London.

Kronlof, C. (1993). *Methods integration: Concepts and case studies.* New York: John Wiley & Sons.

Laurel, B. (1992). Placeholder. Retrieved from www.tauzero.com/Brenda_ Laurel/Placeholder/ Placeholder.html

Lauria, R. (1997). Virtual reality as a metaphysical testbed. *Journal of Computer Mediated Communication, 3*(2). Retrieved from jcmc.huji.ac.il/vol3/issue2/

Lindley, C., Knack, F., Clark, A., Mitchel, G., & Fencott, C. (2001). New media semiotics—computation and aesthetic function. *Proceedings of COSIGN 2001,* Amsterdam. Retrieved from www.kinonet.com/conferences/cosign2001/

Loftin, R.B., & Kenney, P.J. (1994). *The use of virtual environments for training the Hubble Space Telescope flight team.* Retrieved from www.vetl.uk/edu/Hubble/virtel.html

Lombard, M., & Ditton, initial. (1997). At the heart of it all: The concept of telepresence. *Journal of Computer Mediated Communication*, *3*(2). Retrieved September 1997 from jcmc.huji.ac.il/vol3/issue2/

McIntosh, P. Course notes on UML/VRML. Retrieved from www.public. asu.edu/~galatin/

Milner, R. (1989). *Communication and concurrency*. Englewood Cliffs, NJ: Prentice-Hall.

Moser, M.A. (1996). *Immersed in technology*. Boston, MA: MIT Press.

Murray, M. (1997). *Hamlet on the Holodeck: The future of narrative in cyberspace*. New York: The Free Press.

Rollings, A., & Adams E. (2003). *Andrew Rollings and Ernest Adams on games design*. New Riders.

Ryan, T. (1999). Beginning level design. Retrieved from www.gamasutra.com

Saltzman, M. (ed.). (1999). *Games design: Secrets of the sages*. Macmillan.

Sánchez-Segura, M.I., Cuadrado, J.J., de Antonio, A., de Amescua, A., & García L. (2003). Adapting traditional software processes to virtual environments development. *Software Practice and Experience, 33*(11). Retrieved from www3.interscience.wiley.com/cgi-bin/jhome/1752

Slater, M. (1999). Co-presence as an amplifier of emotion. *Proceedings of the Second International Workshop on Presence,* University of Essex, UK. Retrieved from www.essex.ac.uk/psychology/tapestries/

Spinney, L. (1998). I had a hunch.... *New Scientist,* (September 5).

Trondstad, R. (2001). Semiotic and nonsemiotic MUD performance. *Proceedings of COSIGN 2001,* CWI, Amsterdam.

Turkle, S. (1995). *Life on the screen: Identity in the age of the Internet.* Phoenix.

UML Version 1.1 Summary. Retrieved from www.rational.com/uml/resources/documentation/summary/

Whitelock, D., Brna, P., & Holland, S. (1996). *What is the value of virtual reality for conceptual learning? Towards a theoretical framework.* Retrieved from www.cbl.leeds.ac.uk/~paul/papers/vrpaper96/VRpaper.html

Workshop on Structured Design of Virtual Environments. (2001). *Proceedings of Web3D Conference*, Paderborn, Germany. Retrieved from www.c-lab.de/web3d/VE-Workshop/index.html

Workshop on Usability Evaluation for Virtual Environments. (1998). De Montforte University. Retrieved from www.crg.cs.nott.ac.uk/research/technologies/evaluation/workshop/workshop.html

This work was previously published in Developing Future Interactive Systems, edited by M.-I Sanchez-Segura, pp. 66-91, copyright 2005 by IGI Publishing, formerly known as Idea Group Publishing (an imprint of IGI Global).