INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films
the text directly from the original or copy submitted. Thus, some thesis and
dissertation copies are in typewriter face, while others may be from any type of
computer printer.

The quality of this reproduction is dependent upon the quality of the
copy submitted. Broken or indistinct print, colored or poor quality illustrations
and photographs, print bleedthrough, substandard margins, and improper
alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript
and there are missing pages, these will be noted. Also, if unauthorized
copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by
sectioning the original, beginning at the upper left-hand corner and continuing
from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced
xerographically in this copy. Higher quality 6" x 9" black and white
photographic prints are available for any photographs or illustrations appearing
in this copy for an additional charge. Contact UMI directly to order.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®
NOTE TO USERS

This reproduction is the best copy available.

UMI
A VIRTUAL REALITY INTERFACE DESIGN (VRID) MODEL AND METHODOLOGY

A dissertation

submitted by

Vildan Tanriverdi

In partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

in

Computer Science

TUFTS UNIVERSITY

May 2001

Adviser: Dr. Robert J.K. Jacob
ABSTRACT

Virtual reality (VR) interfaces need to support a richer variety and more complex types of human-computer interactions, objects, behaviors, and communications than those observed in conventional interfaces. Therefore, design and development of VR interfaces is a challenging process that is currently more of an art than a science. There is need for design models and methodologies that can guide designers in: a) thinking comprehensively about the overall design of the VR interface; b) decomposing the design task into smaller, conceptually distinct, and easier tasks; and c) communicating the structure of the design to software developers. In this thesis, we develop a virtual reality interface design (VRID) model, and an associated VRID methodology to guide the design of virtual reality interfaces. The VRID model aims to provide guidance to designers as to what they should be doing in the design and why, while the VRID methodology provides procedural steps and guidelines as to how the design model can be applied methodically. The VRID model uses a multi-component object architecture to help designers specify graphical features, behaviors, interactions, and communication mechanisms of interface objects. The VRID methodology guides designers in applying the VRID model during the design process and in generating design specifications at high and low levels of abstraction, and it facilitates iterations and refinements between the two levels until a conceptually sound and practically implementable design emerges. We test validity, usability, and usefulness of the VRID model and methodology in two ways. First, we apply the VRID model and methodology in the design of a well-known VR interface, and discuss the advantages of using VRID vis-a-vis other design models and methodologies. Second, we conduct an experimental user study in which two groups of
designers design a given VR interface. One group uses the VRID while the other group, the control group, uses object oriented (OO) design model and methodology for designing the same VR interface. We compare the two models and methodologies along three dimensions: a) performance of designers; b) satisfaction of designers with VRID and OO models and methodologies; and c) quality of the resulting designs. We find that the VRID model and methodology are superior than the OO model and methodology in all three dimensions.
ACKNOWLEDGMENTS

I would like to thank to

Dr. Robert J.K. Jacob, my advisor and the chair of the dissertation committee, for his insights, inspiration and advices

Dr. Stephen A. Morrison, Dr. James G. Schmolze and Dr. Charles Wiecha, members of the dissertation committee, for their helpful suggestions and comments

Participants of the user studies for their time and feedback

My parents for their support through my entire life, and particularly

My husband, Hüseyin Tanriverdi, for his love, support and encouragement.

This work was supported by National Science Foundation Grant IRI-9625573 and Office of Naval Research Grant N00014-95-1-1099. I gratefully acknowledge their support.
# TABLE OF CONTENTS

**CHAPTER 1: INTRODUCTION** .......................................................... 1

**CHAPTER 2: BACKGROUND AND RELATED WORK** .......................... 7
DISTINCTIVE CHARACTERISTICS OF VR INTERFACES ............................ 7
THE NEED FOR DESIGN MODELS AND METHODOLOGIES ...................... 10
  Distinguishing between a design model and a design methodology .... 10
  Distinguishing between high and low levels of abstraction ............. 11
MODELS AND METHODOLOGIES THAT GUIDE DESIGN OF
CONVENTIONAL INTERFACES ...................................................... 12
RELATED WORK ON THE DESIGN OF VR INTERFACES ......................... 17

**CHAPTER 3: VRID MODEL** ............................................................ 22
COMPONENTS OF A VIRTUAL REALITY SYSTEM .................................. 22
THE MODEL .............................................................................. 23

**CHAPTER 4: VRID METHODOLOGY** ............................................. 32
HIGH-LEVEL (HL) DESIGN PHASE .................................................. 34
  HL1: Identifying data elements ............................................. 34
  HL2: Identifying objects .................................................... 35
  HL3: Modeling the objects ................................................ 37
LOW-LEVEL (LL) DESIGN ............................................................. 41
  LL1: Graphics .................................................................... 42
  LL2: Behaviors .................................................................. 44
  LL3: Interactions ................................................................ 44
  LL4: Internal communications (mediator) ............................. 45
  LL5: External communications ......................................... 46

**CHAPTER 5: GORILLA EXHIBIT EXAMPLE** .................................. 47
APPLYING VRID METHODOLOGY TO THE GORILLA EXHIBIT: ............ 48
  High Level Design Phase: .................................................. 48
  Low Level Design Phase: .................................................. 61
DISCUSSION ............................................................................ 70

**CHAPTER 6: ESTABLISHING THE VALIDITY OF THE VRID
MODEL AND METHODOLOGY: AN EMPIRICAL STUDY** ............... 74
METHOD .................................................................................. 74
  Independent variable ....................................................... 74
  Dependent variables ....................................................... 75
  Participants ....................................................................... 77
  Procedure ......................................................................... 78
RESULTS .................................................................................. 80
  Performance results .......................................................... 81
  Designer satisfaction with the VRID and OO model and methodology 81
CHAPTER 7: CONCLUSIONS AND FUTURE WORK

APPENDIX A: EVALUATIONS OF DESIGN MODEL AND METHODOLOGIES

1. THE VRID MODEL AND METHODOLOGY
   Questionnaire 1 ................................................................. 91
   Questionnaire 2 ................................................................. 93
   The Results of Questionnaire 1 .............................................. 97

2. OBJECT ORIENTED DESIGN MODEL AND METHODOLOGY
   Questionnaire 1 ................................................................. 98
   Questionnaire 2 ................................................................. 99

APPENDIX B: EVALUATIONS OF DESIGN SOLUTIONS

1. THE VRID MODEL AND METHODOLOGY
   Evaluation Questionnaire-1 .................................................. 103
   Evaluation Questionnaire-2 .................................................. 106

2. OBJECT ORIENTED DESIGN MODEL AND METHODOLOGY
   Evaluation Questionnaire-1 .................................................. 109
   Evaluation Questionnaire-2 .................................................. 111

REFERENCES ................................................................................. 114
LIST OF TABLES

Table 2.1 Comparative characteristics of conventional and VR interfaces ..... 9
Table 2.2 Characteristics of existing design models and methodologies .......... 15
Table 2.3 Previous research on behavioral and interaction characteristics of VR interfaces ................................................................. 18
Table 4.1 The list of languages, techniques and frameworks that can be used in the low level design phase of the VRID methodology ......................... 43
Table 6.1 Characteristics of the participants ...................................................... 77
Table 6.2 Designers’ satisfaction with VRID and OO models and methodologies ................................................................. 82
Table 6.3 Software developer’s satisfaction with designs built with VRID and OO models and methodologies ................................................................. 83
Table 6.4 Ease of modifiability of designs built with VRID and OO models and methodologies ................................................................. 84
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Components of a VR system</td>
<td>22</td>
</tr>
<tr>
<td>3.2</td>
<td>Multi-component object architecture used in the VRID model</td>
<td>24</td>
</tr>
<tr>
<td>4.1</td>
<td>Outline of the VRID methodology</td>
<td>33</td>
</tr>
<tr>
<td>4.2</td>
<td>Description of the virtual surgery system</td>
<td>34</td>
</tr>
<tr>
<td>4.3</td>
<td>PMIW representation of bleeding behavior</td>
<td>45</td>
</tr>
<tr>
<td>4.4</td>
<td>PMIW representation of puncturing interaction</td>
<td>45</td>
</tr>
<tr>
<td>5.1</td>
<td>Description of Gorilla Exhibit</td>
<td>47</td>
</tr>
<tr>
<td>5.2</td>
<td>Identification of data elements</td>
<td>48</td>
</tr>
<tr>
<td>5.3</td>
<td>Identification of objects</td>
<td>49</td>
</tr>
<tr>
<td>5.4</td>
<td>High level graphical specifications</td>
<td>50</td>
</tr>
<tr>
<td>5.5</td>
<td>High level behavioral specifications</td>
<td>53</td>
</tr>
<tr>
<td>5.6</td>
<td>High level interaction specifications</td>
<td>56</td>
</tr>
<tr>
<td>5.7</td>
<td>High level internal communications specifications</td>
<td>58</td>
</tr>
<tr>
<td>5.8</td>
<td>High level external communications specifications</td>
<td>60</td>
</tr>
<tr>
<td>5.9</td>
<td>Low level graphical specifications</td>
<td>62</td>
</tr>
<tr>
<td>5.10</td>
<td>Low level behavioral specifications with PMIW</td>
<td>66</td>
</tr>
<tr>
<td>5.11</td>
<td>Low level interaction specifications with PMIW</td>
<td>67</td>
</tr>
<tr>
<td>5.12</td>
<td>Low level internal communications specifications</td>
<td>68</td>
</tr>
<tr>
<td>5.13</td>
<td>Low level external communications specifications</td>
<td>69</td>
</tr>
<tr>
<td>6.1</td>
<td>VR interface design task</td>
<td>80</td>
</tr>
</tbody>
</table>
CHAPTER 1: INTRODUCTION

Virtual reality (VR) technologies have experienced significant improvements since the introduction of the first VR system in 1963 (Sutherland 1963). Advances in computer graphics and hardware technology now enable virtual environments to display realistic, immersive graphic worlds. VR is seen as a promising platform for development of new applications in medicine, entertainment, visualization of scientific and business information, and in other areas (Anderson, Barrus et al. 1995; Hodges, RothBaum et al. 1995; Koller, Lindstrom et al. 1995; Risch, May et al. 1996; State, Livingston et al. 1996; Strickland 1996; Allison, Wills et al. 1997; Satava and Robb 1997; Fuchs, Livingston et al. 1998; Rizzo, Buckwalter et al. 2000; Sorid and Moore 2000). Despite their potential advantages, however, we do not yet see widespread development, adoption and usage of VR applications in practice. VR applications have only recently begun to emerge out of pilot studies and prototypes in research labs and to move into production stage for applications in practice (Brooks 1999).

The lack of proliferation of VR applications can be attributed partly to the challenges of building VR applications (Astheimer 1999; Brooks 1999). In particular, interfaces of the VR applications, which provide communication between the user and the application, are more complex and challenging to design compared to interfaces of conventional desktop based applications (Bryson 1996; Bowman 1999). VR interfaces need to support a richer variety and more complex types of human-computer interactions, objects, behaviors, and communications than those observed in conventional interfaces.
There is a growing body of work that addresses the challenges of designing VR interfaces. Some researchers have developed frameworks for modeling interactions (Kessler 1999) and behaviors (Blumberg and Galyean 1995; Cremer, Kearney et al. 1995), and languages to specify low-level details of behaviors (Green and Halliday 1996; Steed and Slater 1996). Others have developed languages which can be used for describing and programming fine-grained aspects of interactions in VR interfaces (Jacob, Deligiannidis et al. 1999; Smith and Duke 1999). Tools for specifying low-level design details and prototyping of VR interfaces are also available (Shaw, Green et al. 1993; Deering 1995; Pausch, Burnette et al. 1995; Billinghurst and Savage 1996).

As the descriptions imply, existing frameworks and tools focus on design and development of specific components of the VR interface such as interactions and behaviors. They serve as building blocks that designers can select and use in different components and phases of the overall design and development task. However, the existing frameworks and tools do not provide high-level guidance to designers in conceptualizing the overall interface design and in systematically going about the design process. In the absence of high-level design models and methodologies that provide conceptual guidance and bring discipline to the design process, the design of VR interfaces remains to be a challenging process that can currently be described as more of an art than a science. There is need for design models and methodologies that can: a) provide conceptual guidance to designers in thinking comprehensively about the overall design of the VR interface; b) decompose the design task into smaller, conceptually
distinct, and easier tasks; and c) communicate specifications of the design to software developers, who are responsible for the coding.

In this thesis, we propose a virtual reality interface design (VRID) model, and an associated VRID methodology that provide support to the design of VR interfaces. The VRID model aims to provide guidance to designers as to what they should be doing in the design and why, while the VRID methodology provides procedural steps and guidelines as to how the design model can be applied methodically. The VRID model uses a multi-component object architecture to help designers specify graphical features, behaviors, interactions, and communication mechanisms of interface objects. It allows designers to: a) think comprehensively about the overall design of the VR interface; b) decompose the design task into smaller, conceptually distinct, and easier tasks; and c) communicate the structure of the design to software developers by providing a common framework and vocabulary.

The VRID methodology guides designers in generating design specifications at high and low levels of abstraction, and facilitates iterations and refinements between the two levels until a conceptually sound and practically implementable design emerges. As Wasserman (1983) points out, the notion of abstraction provides concentration at some level of generalization. At a high level of abstraction, the VRID methodology aims to help designers to concentrate on understanding design requirements and describing design specifications using the language of the interface domain. At a lower level of abstraction,
the methodology aims to help designers to convert the design specifications into procedural details using implementation oriented terminology.

In Chapter 2, we identify distinctive characteristics of VR interfaces that make them more difficult to design and implement compared to conventional interfaces. We review existing design models and methodologies, which were developed for design of conventional interfaces, to see whether and how they address the distinctive needs of VR interface design. We also review relevant work on the design of VR interfaces to assess which components of the overall VR interface design task they address. By doing so, we identify the existing work that can serve as building blocks in the high-level design model and methodology that we seek to develop in this thesis.

In Chapter 3, we introduce the VRID model. We explain components of the model, and provide justification for their inclusion in the model.

In Chapter 4, we introduce the VRID methodology. We describe steps of the methodology, and illustrate how they can guide designers in going about the design of VR interfaces.

In Chapter 5, we demonstrate the applicability of the VRID model and methodology. We use them in designing the interface of “Virtual Reality Gorilla Exhibit,” which is a well-known VR system built by Allison et al. at Georgia Institute of Technology (Allison, Wills et al. 1997). At each step during the design of this VR interface, we compare and
contrast the advantages and disadvantages of using the VRID model and methodology vis-a-vis the design model and methodologies built for conventional interfaces. By doing so, we theoretically establish the validity, usability, and usefulness of the VRID model and methodology.

In Chapter 6, we provide empirical evidence for the validity, usability, and usefulness of the VRID model and methodology. We present an experimental study in which a group of designers designed a VR interface using the VRID model and methodology, and a second group of designers, the control group, used object oriented methodology for designing the same VR interface. We compare the two methodologies by assessing the performance of the two groups of designers, their satisfaction with the respective models and methodologies that they used, and by assessing the quality of the resulting designs. With this study, we provide empirical evidence, which shows that the VRID model and methodology are comparable to or superior than the object oriented methodology in guiding the design of VR interfaces. Designers who used the VRID model and methodology completed the design faster than the designers who used the OO model and methodology. Designers found VRID easier to learn. VRID provided better guidance in decomposing the overall design task into smaller, less complex, and conceptually distinct parts. It also provided better guidance in making design choices and tradeoffs. Overall, designers were more satisfied with the VRID than the OO model and methodology. Software developers, who were responsible for coding the VR interface based on the designs produced by VRID and OO models and methodologies, were also more satisfied with the designs built with VRID. VRID designs were easier to understand. They
provided a higher-level view of the overall coding task for the developers. They also achieved a nice balance between high-level and low-level design specifications, and provided sufficient detail for doing the actual coding. Software developers found the components of the VRID designs more reusable compared to those of the OO designs. VRID designs were also easier to modify. For a given modification task, software developers estimated that the VRID designs would take less time and be easier to modify compared to the OO designs. Overall, software developers found the designs built with VRID model and methodology more successful than the designs built with the OO model and methodology.

In Chapter 7, we discuss the implications of this thesis work for researchers and practitioners. We conclude the work by providing directions for future research in the development of design model and methodologies for VR interfaces.
CHAPTER 2: BACKGROUND AND RELATED WORK

In this chapter, we begin by identifying distinctive characteristics of VR applications that make them more complex and challenging to design compared to interfaces of conventional desktop based applications. Then, we review existing design models and methodologies to explain why they are inadequate for meeting the needs of VR interface design.

DISTINCTIVE CHARACTERISTICS OF VR INTERFACES

Distinctive characteristics of VR interfaces can be grouped into three categories: a) visual characteristics; b) behavioral characteristics; and c) interaction characteristics.

Visual characteristics. VR interfaces predominantly use 3D in their graphical displays. Virtual environments aim to provide users with realistic environments. The sense of "being there" is important especially in the immersive virtual environments. To be able to provide the sense of "being there," VR interfaces heavily use 3D graphical displays. Conventional interfaces, on the other hand, predominantly use 2D graphical displays. In addition to exhibiting differences in graphical displays, VR interfaces also exhibit differences in the types of interface objects they support. VR interfaces typically contain both physical and virtual objects (i.e. computer generated). Physical and virtual objects coexist and exchange information with each other in VR interfaces, especially in the case of "augmented reality" systems (Azuma 1997). Conventional interfaces, however, do not usually contain physical objects. They typically contain virtual, computer-generated objects.
**Behavioral characteristics.** Behavioral characteristics refer to dynamism of objects in the interface. In conventional interfaces, objects usually exhibit passive behaviors. In general, they have predetermined behaviors that are activated in response to an action from the user. Due to passive nature of objects, communication patterns among the objects are usually deterministic and do not present particular challenges to implement. In the case of VR interfaces, objects may present more complex behaviors and communication patterns. VR applications seek to present virtual worlds that contain both real world-like objects and magical objects that do not exist in the real world. Many real world and magical objects are able to change their own states without user’s intervention. In order to create virtual worlds containing such objects, VR interfaces need to support not only the passive objects but also the autonomous objects. Unlike passive objects, autonomous objects can change their own states, they can communicate with each other, and hence, they may affect each other’s behaviors and communication patterns. That is why communication patterns among objects are more complex and non-deterministic in VR interfaces.

**Interaction characteristics.** Explicit user commands are the defining characteristics of conventional user interfaces (Nielsen 1993). Interactions based on explicit commands require users to command the computer explicitly, e.g., by typing delete command or dragging something into a trash can in order to delete it. VR interfaces offer users more ways of interacting with computers. In addition to explicit commands, VR interfaces also support implicit command style interactions, which are also known as non-command
based interactions. Since, VR interface allow users to use their natural skills such as arm, hand, head, or eye movements, interactions become more natural and easier to use. In implicit command style interaction, computer observes and interprets user actions instead of waiting for explicit commands. VR is considered to be the ultimate example of non-command based interfaces that allows the user to move about in the same way as in the physical world (Nielsen 1993). The increasing variety and complexity of interactions to be supported by VR applications constitutes another challenge for the design and implementation of VR interfaces.

We summarize the differences between characteristics of conventional and VR interfaces in Table 2.1. As the table indicates, VR interfaces are more complex than conventional interfaces.

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Conventional interfaces</th>
<th>VR interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object graphics</td>
<td>Mainly 2D</td>
<td>Mainly 3D</td>
</tr>
<tr>
<td>Object types</td>
<td>Mainly virtual objects</td>
<td>Both virtual and physical objects</td>
</tr>
<tr>
<td>Object dynamism</td>
<td>Mainly passive objects</td>
<td>Both passive and active objects</td>
</tr>
<tr>
<td>Communication patterns</td>
<td>Mainly simple</td>
<td>Mainly complex</td>
</tr>
<tr>
<td>Human-computer interactions</td>
<td>Mainly explicit</td>
<td>Both explicit and implicit</td>
</tr>
</tbody>
</table>

Table 2.1 Comparative characteristics of conventional and VR interfaces
THE NEED FOR DESIGN MODELS AND METHODOLOGIES

The increased complexity and variety of graphical representations, object types, object dynamism, object communications and human-computer interactions bring new challenges to the design of VR interfaces. Designers need design models and design methodologies for meeting the challenge. We make a distinction between a design model and a design methodology because they provide different kinds of support and guidance to designers in the design process.

Distinguishing between a design model and a design methodology

A design model provides conceptual guidance to designers in thinking comprehensively about the overall design. A major goal of modeling is to deal with systems that are too complex to understand directly (Rumbaugh, Blaha et al. 1991). Design models reduce complexity by separating out a small number of important things to deal with at a time (Rumbaugh, Blaha et al. 1991). For example, our review above indicates that a designer has to deal with a variety of issues in designing a VR interface such as object graphics, object types, object dynamism, object communications, and human-computer interactions. There is a lot of complexity associated with the VR interface design. However, it is not clear how the designer should conceptualize the VR interface to deal with these issues. A design model should provide a theory or a framework through which the designer can conceptualize and structure the VR interface and reduce the complexity. Since VR interfaces are more complex and challenging to design compared to interfaces of conventional desktop based applications (Bryson 1996; Bowman 1999), a VR interface design model should provide sufficient conceptual support for reducing this complexity.
A design methodology, on the other hand, provides a set of steps and rules that help the designer to go about applying the design model to the problem at hand in a systematic manner. It helps the designer to decompose the design task into smaller, conceptually distinct, and easier tasks in systematic ways; and to represent and communicate the design specifications in ways that can help software developers to easily convert them into software code. The design methodology should present clear steps for applying the design model to the VR interface design problem, and also help designers in selecting and using techniques and notation associated with each step (Rumbaugh, Blaha et al. 1991).

Distinguishing between high and low levels of abstraction

We also make a distinction between the support provided by design models and methodologies at high and low levels of abstraction. Abstraction can be defined as the selective examination of certain aspects of a problem (Rumbaugh, Blaha et al. 1991). The goal of abstraction is to isolate those aspects that are important for some purpose and suppress those aspects that are unimportant (Rumbaugh, Blaha et al. 1991). We define high-level abstraction of a VR interface design as a broad view of the design that isolates conceptual design components and suppresses the implementation-oriented details. Low-level abstraction of a design, on the other hand, can be defined as a detailed view of the design that isolates implementation-oriented details. It is important to make the distinction between the high and low levels of abstraction in order to reduce the complexity associated with VR interface design. At a high-level of abstraction, the designer should be able to focus only on the conceptual design of interactions, object
dynamism, communication patterns, etc. rather than worrying about which tools, techniques, languages, etc. are available for their implementation and how they should be used.

So far, we have reviewed the distinctive characteristics of VR interfaces and argued that designers need models and methodologies for dealing with the complexity introduced into the design by these characteristics. We also explained that it is important to make a distinction between design models and design methodologies, and between the support they provide at high and low levels of abstraction. In the following sections, we review existing design models and methodologies to see whether and how they address the distinctive characteristics of VR interfaces. We also examine whether the existing approaches distinguish between design models and methodologies and between high and low levels of abstraction.

MODELS AND METHODOLOGIES THAT GUIDE DESIGN OF CONVENTIONAL INTERFACES

A well-known user interface design model is the four level approach developed by Foley and colleagues (Foley and Wallace 1974; Foley, van Dam et al. 1996). This model describes the user interface as a medium that provides dialogue between the user and the computer. It builds on the analogy of human communication, which is based on the use of natural languages. The four levels of the model are organized based on the meaning and the form of the dialogue between the user and the computer. The levels focus mostly on specifications of user interactions using explicit commands. This approach works well
for command language and GUI style interfaces. But it is not sufficient to meet VR interface needs such as implicit interactions, object dynamism, and physical objects. Communication among objects is not sufficiently addressed either.

Another relevant user interface design model is the Command Language Grammar (CLG) developed by Moran (1981). It provides designers a model to describe and design command language interfaces. It aims to improve collaboration between users and designers by describing the interface as the user sees and understands it. As in Foley’s model, CLG divides the interface design into levels. But specifications of the levels are more formal and detailed in CLG. Although CLG works well in command language interfaces, its applicability to VR interfaces is limited. Object dynamism, interactions using implicit commands, physical objects, and communication patterns among objects are out of the scope of this model.

A third related interface design model is Shneiderman’s Object-Action Interface (OAI) model (Shneiderman 1998). It is developed particularly for design of GUI (graphical user interface) style interfaces. In order to meet the needs of GUI style interfaces, the OAI model emphasizes the importance of visual representations of objects and their actions. This model focuses on explicit command style interactions using direct manipulations, and keeps the amount of syntax small in interaction specifications. However, OAI does not address the distinctive characteristics of VR interfaces such as object dynamism, interactions using implicit commands, physical objects, and communication patterns among objects.
In addition to the design models and methodologies that are devoted specifically to the
design of user interfaces, designers may also choose to use general-purpose design
models and methodologies that have been proposed for software development. Object
oriented design models and methodologies such as Booch's Model (Booch 1991) and
Rumbaugh's Object Modeling Technique (OMT) (Rumbaugh, Blaha et al. 1991), and
classical methodologies such as Structured Analysis and Design (Yourdon and
Constantine 1979) and Jackson Structured Development (Jackson 1983) may provide
helpful guidelines for designing interfaces. However, these models and methodologies
provide support for design of software systems in general. They do not provide
conceptual guidance for addressing the specific needs of the VR interfaces such as
interactions between the user and the computer, and communications with underlying
application of the system being developed. Hence, they are too generic to provide
conceptual guidance to designers in thinking comprehensively about the overall design of
the VR interface.

In Table 2.2, we provide a summary of the design models and methodologies that we
reviewed so far. This table assesses the existing work along two dimensions: a) model
and methodology related features; and b) support provided for distinctive characteristics
of VR interfaces. In the first dimension, we examine whether a given approach provides
support to designers at high or low levels of abstraction. We also examine whether the
approach makes a distinction between design model and methodology. Finally, we
identify the type of interface predominately supported by the approach. In the second
dimension, we assess whether the approach addresses the distinctive characteristics of VR interfaces.

<table>
<thead>
<tr>
<th>Design model and methodology</th>
<th>Model and method related features</th>
<th>Support for distinctive characteristics of VR interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Level of abstraction</td>
<td>Model and method separation?</td>
</tr>
<tr>
<td>Four level model</td>
<td>High and low level</td>
<td>No</td>
</tr>
<tr>
<td>CLG model</td>
<td>High and low level</td>
<td>No</td>
</tr>
<tr>
<td>OAI model</td>
<td>High and low level</td>
<td>No</td>
</tr>
<tr>
<td>OO models</td>
<td>High and low level</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 2.2 Characteristics of existing design models and methodologies

As Table 2.2 shows, existing models provide some support at both high and low levels of abstraction. However, the degree of support provided within each level of abstraction varies across models. For example, the four level model and the CLG focus mainly on syntactic details at low levels of abstraction. A major reason for their focus on low levels of abstraction is the type of interface they seek to support. Both the four level model and the CLG were developed to support command language interfaces in which user commands require syntactic details such as keys of the hardware devices that are used in forming the commands. Therefore, these models focus on device dependent details and
specification of user commands at low levels of abstraction. Since low level syntactic
details disappear in GUI style interfaces (Shneiderman 1998), low level abstraction of
OAI contains specification of user actions in terms of points and clicks on interface
objects without requiring details of underlying hardware. In VR interfaces, the
importance of low level syntactic details diminishes even further (Nielsen 1993).
Therefore, models and methodologies developed for VR interface design should put more
emphasis on high level design.

With the exception of OO models, existing approaches do not make a distinction between
design models and methodologies. They are particularly weak in providing
methodological support. They describe how to apply the design model only implicitly.
We believe that separating the design model and methodology, and explaining the steps
and rules for applying the design model are critical for reducing the complexity
associated with the design of VR interfaces.

As the second part of Table 2.2 indicates, none of the design models and methodologies
that have been developed for conventional interfaces adequately meet the distinctive
needs of the VR interfaces.
RELATED WORK ON THE DESIGN OF VR INTERFACES

In this section, we review relevant work on the design of VR interfaces to assess which components of the overall VR interface design task they address. We had categorized distinctive characteristics of VR interfaces into visual, behavioral, and interaction characteristics. Since the introduction of the first VR system in early sixties, researchers conducted various studies to provide more realistic and immersive interfaces for VR systems. The primary focus of the previous work has been on visual characteristics of VR interfaces and presentation technology that is required to create immersive experiences (Cremer, Kearney et al. 1995). Relatively little research has been devoted to behavioral and interaction characteristics of VR interfaces (Cremer, Kearney et al. 1995; Bowman 1999). In this section, we focus mainly on identifying previous work on the behavioral and interaction characteristics of VR interfaces. We do not review the work on visual characteristics and graphical representations of VR interfaces because we assume that these issues are typically the responsibility of graphical designers rather than VR interface designers. In developing our design model and methodology, we will address the visual characteristics of VR interfaces only to facilitate communications between interface designers and graphical designers.

Table 2.3 provides a summary of the previous work addressing behavioral and interaction characteristics of VR interfaces.
<table>
<thead>
<tr>
<th>Frameworks and models</th>
<th>Languages</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>Behavioral</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blumberg and Galyean 1995</td>
<td>Steed and Slater 1996</td>
<td></td>
</tr>
<tr>
<td>Tu and Terzopoulos 1994</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gobbetti and Balaguer 1993</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interaction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kessler 1999</td>
<td>Jacob, Deligiannis et al. 1999</td>
<td>Interaction techniques: Bowman and Hodges 1997</td>
</tr>
<tr>
<td>Lewis, Koved et al. 1991</td>
<td>Smith and Duke 1999</td>
<td>Liang and Green 1993</td>
</tr>
<tr>
<td>Gobbetti and Balaguer 1993</td>
<td></td>
<td>Poupyrev, Billinghurst et al. 1996</td>
</tr>
<tr>
<td>Bowman 1999</td>
<td></td>
<td>Stoakley, Conway et al. 1995</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tanriverdi and Jacob 2000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Wloka and Greenfield 1995</td>
</tr>
</tbody>
</table>

Table 2.3 Previous research on behavioral and interaction characteristics of VR interfaces

A major study on the specification of object behaviors in VR interfaces is HCSM, a framework based on *Hierarchical, Communicating and Concurrent State Machines* (Cremer, Kearney et al. 1995). HCSM aims to specify autonomous behaviors of objects in real time virtual environments. A similar architecture built by Blumberg and Galyean describes behaviors of autonomous creatures and how to coordinate these behaviors using external controls (Blumberg and Galyean 1995). Studies by Tu and Terzopoulos on modeling and simulation of living systems provide a systematic approach to specification of individual and group behaviors of creatures (Tu and Terzopoulos 1994; Terzopoulos 1999). Another study by Gobbetti and Balaguer proposes a dynamic model that provides
support for composing sophisticated behaviors from simpler state information and to specify general dependencies between objects (Gobbetti and Balaguer 1993). Researchers have also developed languages to facilitate specification of behaviors. Two major examples are the behavioral language that is used by MR Toolkit (Green and Halliday 1996) and Steed’s dataflow based language (Steed and Slater 1996). Furthermore, behavioral library proposed by Stansfield and colleagues is one of the major attempts to simplify specification of behaviors by taking advantage of reusability (Stansfield, Shawver et al. 1995).

As a result of the increasing object dynamism and autonomy in VR interfaces, complex communication patterns among objects have become significant parts of the design and development process. HCSM supports communication among objects by providing specialized objects that orchestrate behaviors of multiple objects. Blumberg and Galyean support communication among objects by sensory system in their architecture, which provides sensing via direct interrogation of other virtual creatures and objects. Similarly, Tu and Terzopoulos’s model provides communication of an object with the rest of the environment via perception component that provides sensory information about the surrounding environment. The model proposed by Gobbetti et al. manages communication among objects using constraints that defines relations among objects.

User interactions in virtual environments have also proved to be a challenging issue in development of successful VR interfaces. User interactions are more complex and have broader scope in VR interfaces than conventional interfaces. VR interfaces offer users to
use variety of their natural skills such as speech, gestures, head, body and eye movements. Researchers have investigated various interaction techniques to enable users to use these natural skills effectively in virtual environments (Liang and Green 1993; Stoakley, Conway et al. 1995; Wloka and Greenfield 1995; Poupyrev, Billinghurst et al. 1996; Bowman and Hodges 1997; Tanriverdi and Jacob 2000). Languages have been developed to facilitate specification of interactions (Jacob, Deligiannidis et al. 1999; Smith and Duke 1999). In addition to the studies that provide low level support for interactions, there have been attempts for higher level support for designing and implementing interactions. The framework developed by Kessler aims to guide developers in designing and implementing interaction techniques, combining with other interaction techniques to produce more complex interactions and reusing them in different applications (Kessler 1999). Another framework by Lewis and colleagues provides an architecture that specifies interaction using a conceptual hierarchy that transforms a specific device into a generic device, or transforms a generic device into an interaction technique, or maps interaction techniques to actions (Lewis, Koved et al. 1991). Bowman's methodology (Bowman 1999) helps developers to design and evaluate interaction techniques for given interaction tasks.

The studies reviewed above provide solutions for individual characteristics and requirements of VR interfaces such the behavioral aspects, interaction aspects or a particular issue within these domains. For a given VR interface description, in which the design task is not formulated in terms of behaviors, interactions, communications, etc., these studies do not provide guidance to the designer as to what to do and how to do it.
They leave it up to the designer to figure out conceptualization of the interface in terms of various types of behaviors, interactions, graphics designs, communications, interactions etc. Once the designer figures out how to conceptualize the interface and how to decompose the overall design task into behaviors, interactions, communications, graphics, etc., she or he can use the individual solutions provided by these studies. In the absence of higher level models that bring together these individual solutions, designers cannot reduce the complexity associated with the design of VR interfaces. Therefore, there is need for design models and methodologies that provide guidance to designers at a higher level of abstraction and interconnect the existing solutions. The VRID model and methodology that we propose in the next two chapters aim to address this need.
CHAPTER 3: VRID MODEL

In this chapter, we introduce the VRID model. In order to circumscribe the boundaries and scope of the VRID model, it is important to clarify how we conceptualize a VR system, and what we mean by a VR interface. Therefore, we start by presenting and defining the major components of a VR system. Then, we proceed to explaining the architecture of our VRID model.

COMPONENTS OF A VIRTUAL REALITY SYSTEM

Building on the Seeheim user interface system architecture (Green 1983), we conceptualize a VR system in terms of three major components: application, interface, and dialog control, as depicted in Figure 3.1.

![Diagram of VR System Components]

Figure 3.1 Components of a VR system

Application component is the VR application itself, which contains features, rules and knowledge defining the logic of the application. Interface component is the front-end through which users and other external entities exchange information with and
manipulate the system. The interface consists of data and objects. Data refers to inputs received from users (or other external entities) whereas objects refer to entities in the interface that have well defined roles and identities. Dialog control enables communication between the application and the interface. Due to conceptual separation, internal details of application and interface components are transparent to each other. This feature allows designers to work on the two components independently. The VRID model and methodology that we propose focus only on the design of the interface component in the VR systems. The design of the application or the dialog control components is beyond the scope of this thesis.

THE MODEL

Building on our review and synthesis of the previous work on VR interface design, we identify object graphics, object behaviors, object interactions and object communications as the key constructs that designers should think about in designing VR interfaces. Therefore, we organize our VRID model around the multi-component object architecture that is depicted in Figure 3.2. The model requires five components for each object in the interface. The goal of specifying graphics, behaviors, communication and interaction components is to address the needs of distinctive characteristics of VR interfaces, mainly visual, behavioral and interaction characteristics, as we described in chapter 2. Mediator component is a new component that we propose for coordinating communications among the other four components of an object. In the following sections, we will explain and justify each of these components in detail.
**Graphics component** is for specifying various graphical representations of interface objects. It covers specification of all graphical models that are needed for computer-generated appearance and animations of the objects. We include the graphics component in the model in order to address the distinctive visual characteristics of VR interfaces. In general, the responsibility for the design of the visual aspects of VR interfaces lies with graphical designers rather than VR interface designers. That is why we do not seek to provide design guidance to VR interfaces designers in visual aspects of VR interfaces. Instead, we treat the graphics component as a black box, and focus only on its outcomes, i.e., graphical models. We aim to provide guidance to VR interface designers in specifying graphical models associated with interface objects and their behaviors. By doing so, we seek to facilitate the communications between VR interface designers and graphical designers.

**Behavior components** are for specifying various types of object behaviors. Interface objects may exhibit "physical" behaviors, which are based on simple physics and easily observable in real world counterparts of the interface objects (e.g., a virtual basketball
exhibiting falling and bouncing off behaviors like a real world basketball). They may also exhibit “magical” behaviors that are rarely observable or not observable in their real world counterparts. For example, a virtual basketball that changes its color and displays player statistics (e.g., ball possession, scores, etc.) when touched by a player, is exhibiting “magical” behaviors, which cannot be observed in a real world basketball. Indeed, objects may exhibit behaviors that consist of a series of physical behaviors, magical behaviors, or a combination of both. For example, a virtual basketball that bounces off the floor and moves up is exhibiting a composite behavior that consists of a series of physical behaviors. A virtual basketball that changes color and displays player statistics when touched by a different player exhibits a composite behavior consisting of a series of magical behaviors. A virtual basketball that moves forward by bouncing on and off between the hand of the player and the floor while displaying the statistics of the player in possession is exhibiting a composite behavior that consists of a combination of physical and magical behaviors. In order to guide design of various types and complexities of object behaviors, we make a conceptual distinction among "physical," "magical," and "composite behaviors" described above, and dedicate a separate component for specifying each of these behaviors in the design model.

**Physical behavior component** is for specifying simple physics changes in an object’s state, which are similar to or the same as changes in the state of the corresponding physical objects in the real world.

**Magical behavior component** is for specifying simple changes in an object’s state that are rarely observable or not observable in the real world.
Composite behavior component is used for specifying more complex behaviors that are made up of a combination of physical and/or magical behaviors. The decision of classifying a behavior as composite or simple is based on the level of details that need to be described in the design. For example, long jump behavior of an athlete can be considered as a composite behavior that consists of running and jumping behaviors. Running or jumping could also be specified as a composite behavior by combining and coordinating simple physical behaviors such as leg and arm movements in various directions. Composite behavior component specifies how simple leg and arm movements should be combined and coordinated to generate a coherent behavior. On the other hand, if the details are not necessary, running and jumping could be considered as simple behaviors.

The distinction among "physical," "magical," and "composite behaviors," allows designers to decompose various types and complexities of behaviors into smaller, and conceptually distinct parts. It also increases reusability of the designs and associated software code. Classification of behaviors into simple and composite behavior categories allows designers to decompose complex behaviors into a series of simple behaviors. Further classification of simple behaviors into physical and magical behavior categories generates a library of simple physical and magical behaviors. Designers can combine these simple behaviors in different ways and sequences to generate new composite behaviors. For example, recognizing running behavior as composite behavior allows designers to decompose the behavior into a series of simple arm and leg movements. The designs and software code developed for simple arm and leg movements can be reused in developing related behaviors such as jumping and walking, which only require different
combinations of the simple arm and leg movements. If running behavior were designed and coded as a standalone behavior, reusability of the resulting design and code would be limited.

Unlike prior research, which focused mainly on presentation technologies (Cremer, Kearney et al. 1995), we devote specific attention to the design of object behaviors because defining object behaviors has typically been a challenge in VR (Green and Halliday 1996). Most virtual environments developed to date remain visually rich, but behaviorally impoverished (Cremer, Kearney et al. 1995). Related work that we reviewed in the second chapter addresses how to specify object behaviors. However, they do not provide high level guidance for decomposing complex behaviors into simpler behaviors, which can be recombined and reused in different ways for generating new behaviors. Our behavior component provides conceptual guidance to designers for reducing the complexity associated with object behaviors, and for increasing the reusability of the resulting behavior components. It achieves these outcomes by having designers to think about and distinguish between physical and magical behaviors and between simple and composite behaviors.

**Interaction component** is used to specify where inputs of the VR system come from and how they change object behaviors. Inputs may come from users or other external entities such as a physical device, or another VR system. The interaction component receives the input, interprets its meaning, decides on implication of the input for object behavior, and communicates with behavioral components to make the desired change in object behavior.
To illustrate how the interface component works, consider a hypothetical educational VR system intended to teach the concept of watering flowers. For simplicity, let the system have a flower and a rain cloud. Initially, the flower is pale and the rain cloud is far away from the flower. When the user grabs and drags the rain cloud over to the flower, it starts raining. The flower smiles and starts blossoming. If the cloud keeps raining for long, the flower frowns and starts fading. In this example, interaction component of the cloud object receives the input from the user. It interprets the meaning of the input as grabbing of the cloud and dragging of it over to the flower. It decides that the implication of this input for the cloud is to initiate the raining behavior. Hence, it communicates with the physical behavior component to initiate the raining behavior. At this point, the flower starts exhibiting smiling and blossoming behaviors.

We adopted the interaction component in the VRID model by building on the previous work on interaction aspects of VR interfaces, what was reviewed in chapter 2. The only difference in our approach is that we start analyzing the interactions and their implications for object behaviors at a high level of abstraction without going into representation tools and details. Previous work typically starts representing the interactions at low levels of abstraction, e.g., by using state transition diagrams. Our approach allows designers to think about the interactions conceptually first, without worrying about representing them with particular tools, which is the focus of the lower level design.

**Mediator component** is for specifying control and coordination mechanisms for communications among other components of the object. The goals are to avoid conflicts
in object behaviors, and to enable loose coupling among components. To achieve these goals, we propose the mediator component by adapting the concept of "mediator design pattern" suggested by Gamma and colleagues (1995). The mediator controls and coordinates all communications within the object. When a component needs to communicate with another component, it sends its message to the mediator rather than sending it directly to the destination. This component enables designers to identify, in advance, which communication requests might lead to conflicts in object behaviors, and to specify how the requests can be managed to avoid the conflicts. Since a component only needs to know about itself and the mediator rather than having to know about all components with which it might communicate, the mediator component also ensures loose coupling between components.

Recall the flower watering example. Assume that the flower is in the state of smiling and blossoming as a result of raining. After a pre-specified amount of time, the VR application sends a time warning to the flower, to indicate that it has received too much water. This warning is received by the communication component, which will be described below. Communication component sends this request to the mediator component that notifies composite behavior component to change the behavior of the flower from smiling and blossoming into frowning and fading.

As an example of conflicting requests, let the object "Sun" be added to the system. When the user grabs and drags the Sun over to the flower, the flower will exhibit "smiling and growing" behaviors. Suppose that the system is about to enter the state of raining for long, which will in turn cause the flower to exhibit "frowning and fading" behaviors. Suppose further that, at the same point in time, the user grabs and drags the Sun over to
the flower. At this point, mediator of the flower will receive two communications requests for the composite behavior component: exhibit "frowning and fading" behavior, and exhibit "smiling and growing" behaviors. If the mediator passes these requests simultaneously to the composite behavior component, an inconsistent behavior may occur since the two requests are conflicting. Designers can avoid such conflicts by specifying how the mediator should prioritize and sequence communications requests that cause the conflicts.

**Communication component** enables external communications of the object with other objects, data elements, or with the application component. In this component, designers need to specify sources of communication inflows into the object, destinations of communications outflows from the object, and the message passing mechanisms between them such as the synchronous, asynchronous, balking, or timeout mechanisms discussed by Booch (1991). Previous work on object behaviors discuss message passing mechanisms among objects during low level specifications of behaviors. The difference in our approach is that we start analyzing the communication needs of objects at a higher level of abstraction, and that we make a distinction between internal and external communications of objects. By using two separate components for specification of internal and external communications mechanisms of objects, we help designers to decompose the complexity associated with design of communications into smaller, conceptually distinct components, which are easier to analyze, design, code, and maintain.
The VRID model synthesizes previous work on VR interface design and proposes a comprehensive set of modeling structures in the form of a multi-component object architecture. While much of the previous work focuses on individual components of the VR interface such as behaviors, interactions, communications, etc., the VRID model covers all aspects of a VR interface comprehensively. Hence it reduces the cognitive load of the designer. For a given VR interface description, the designer using the VRID model can immediately focus on communication needs, behaviors, interactions and graphical representations of objects. In the absence of the VRID model, the designer would have to figure out where to begin and how to model the interface. In other words, the VRID model serves as a framework for conceptualizing the VR interface. It helps designers to see clearly which issues and decisions are involved in VR interface design, and why. It allows designers to think comprehensively about various types of human-computer interactions, objects, behaviors, and communications that need to be supported by VR interfaces. It also provides a common framework and vocabulary, which can enhance communication and collaboration among users, designers and software developers involved in development of VR interfaces. Flexibility of the model in terms of types of objects, behaviors, interfaces, and communications supported makes it applicable in a broad range of VR interfaces.
CHAPTER 4: VRID METHODOLOGY

In this chapter, we propose a methodology for systematically applying the VRID model in VR interface design. We conceptualize design of a VR interface as an iterative process in which requirements for the interface are translated into design specifications that can be implemented by software developers. We divide the design process into high-level and low-level design phases. In the high-level design phase, the goal is to specify a design solution, at a high-level of abstraction, using the multi component object architecture as a conceptual guidance. The input of the high-level design phase is a functional description of the VR interface. The output is a high-level representation of data elements and objects in the interface. Graphical models, behaviors, interactions, and internal and external communication characteristics of interface objects are identified and defined at a high level of abstraction. This output becomes the input of the low-level design phase. In the low-level design phase, the goal is to provide fine-grained details of the high-level representations, and to provide procedural details as to how they will be formally represented. The outcome of low-level design is a set of design specifications, which are represented in formal, implementation-oriented terminology, and ready to be implemented by software developers. We take a top-down approach to the design process by going from high-level abstractions to lower level details. However, this is not a linear process. It requires iterations between the high-level and low-level design phases, and reciprocal refinements at both levels of abstraction until a conceptually sound and practically implementable design emerges. The steps of the VRID methodology are shown in Figure 4.1.
I. High Level (HL) Design Phase
  HL 1 - Identifying data elements
  HL 2 - Identifying objects
  HL 3 - Modeling the objects
    HL 3.1 - Graphics
    HL 3.2 - Behaviors
    HL 3.3 - Interactions
    HL 3.4 - Internal communications (mediator)
    HL 3.5 - External communications

II. Low Level (LL) Design Phase
  LL1 - Graphics
  LL2 - Behaviors
  LL3 - Interactions
  LL4 - Internal communications (mediator)
  LL5 - External communications

Figure 4.1 Outline of the VRID methodology

In the following sections, we explain step-by-step details of each phase of the VRID methodology. We use an example, which runs throughout the chapter, to illustrate how the VRID model and methodology are applied in developing a VR interface design. The example, which is described in Figure 4.2, is a hypothetical virtual surgery system inspired by and adapted from the descriptions given in prior studies (State, Livingston et al. 1996; Fuchs, Livingston et al. 1998; Sorid and Moore 2000). The example needs to be simple enough to avoid burdening the reader with too much complexity, yet sufficiently large in scope and functionality to illustrate key features and benefits of the VRID model and methodology.
Consider an augmented reality system developed for training surgeons. It includes a virtual patient body and a physical biopsy needle. The surgeon wears a head-mounted display, and uses the needle to interact with the patient. A Polhemus is attached to the needle to communicate 3D coordinates of the needle to the VR system. Coordinate data is used to interpret actions of the surgeon. Abdominal area of the body is open, and organs, nerves, vessels, and muscles are visible. When the surgeon punctures an organ with the needle, it starts bleeding. The surgeon can see status of operation by prodding the organ. When prodded, the organ shows bleeding, and the status of surrounding nerves, vessels, and muscles by highlighting each of them with a unique color. This allows the surgeon to identify the source of bleeding, and to determine if he has correctly punctured the intended point.

**Figure 4.2 Description of the virtual surgery system**

**HIGH-LEVEL (HL) DESIGN PHASE**

High-level design phase consists of three major steps:

- Step1 - (HL1) Identifying data elements
- Step2 - (HL2) Identifying objects
- Step3 - (HL3) Modeling the objects
  - Step 3.1 - (HL3.1) Graphics
  - Step 3.2 - (HL3.2) Behaviors
  - Step 3.3 - (HL3.3) Interactions
  - Step 3.4 - (HL3.4) Internal communications (mediator)
  - Step 3.5 – (HL3.5) External communications

**HL1: Identifying data elements**

The role of data elements is to enable communication between VR interface and entities that are external to the VR system. The goal of the first step is to identify data inflows coming into the VR interface. The interface can receive data from three sources: a) users,
b) physical devices; and c) other VR systems. Designer should analyze the description of
the VR interface to identify the data inflows. Identification of data elements is a relatively
simple design task, which does not require deliberations at different levels of abstraction.
We include this task in the high-level design phase in order to enable designers to
understand and define data inputs of the VR interface early in the design process.

In the virtual surgery example, the only data element is the 3D coordinates of the needle
communicated to the interface by the Polhemus.

**HL2: Identifying objects**

In this step, the goal is to identify objects that have well defined roles and identities in the
interface. This step involves: a) identifying potential objects mentioned in the interface
description; b) deciding on legitimate objects; and c) distinguishing between virtual and
physical objects. In parts (a) and (b), designers can use the object-oriented analysis and
design guidelines provided for identification of potential objects and selection of
legitimate objects (Booch 1991; Coad and Yourdon 1991; Rumbaugh, Blaha et al. 1991).
Objects can be single objects or aggregate objects that consists of multiple objects, e.g.
the patient body comprising of organ nerve, vessel and muscle components. In part (c),
virtual objects are those entities that need to be modeled and generated by the computer.
Physical objects are physical entities that interact with the VR system. Physical objects
may or may not require modeling. If they are capable of coexisting and exchanging data
with the VR interface, they do not require modeling. For example, the biopsy needle in
our virtual surgery example is capable of sending data to the VR interface through the
Polhemus. Hence, it should be identified as a physical object. Physical objects that
exhibit magical behaviors need to be identified and modeled as virtual objects. For example, a biopsy needle, which is capable of melting down and disappearing when the surgeon makes a wrong move, is exhibiting a magical behavior. This behavior is only possible through computer generation since no physical biopsy needle is capable of exhibiting this behavior. Therefore, such objects should be modeled as virtual objects.

Although entities that are identified as "physical objects" do not require modeling, in some cases, there may be benefits in specifying and modeling them as virtual objects. Many physical objects (e.g., buttons, dials, joysticks, steering wheels, etc.) lack natural mappings that facilitate interaction tasks. They can be difficult to find and use while wearing a head-mounted display (Mine 1995). Modeling them as "virtual objects" may overcome some of the limitations because just about anything that can be imagined can be implemented as a virtual object (Mine 1995).

*In the virtual surgery example, potential objects are biopsy needle, patient body, organs, nerves, vessels, and muscles. In parts (a) and (b), biopsy needle and patient body can be identified as legitimate objects using the general guidelines of object-oriented analysis and design. Patient body is an aggregate object comprising of organ, nerve, vessel, and muscle components. In part (c), the needle can be identified as a physical object because it is capable of coexisting and exchanging data with the VR interface. The patient body should be identified as a virtual object because it exhibits magical behaviors such as highlighting nerves, vessels, and muscles with unique colors.*
HL3: Modeling the objects

In the remainder of the design, we are no longer concerned with entities that are identified as physical objects because they do not require modeling, and their inputs to the VR system had already been identified as data elements in HL1. Therefore, the goal in this step is to model the virtual objects identified in HL2. Modeling of virtual objects involves specification of: a) graphical models; b) behaviors; c) interactions; d) internal communication characteristics; and e) external communication characteristics of the objects. Aggregate objects that consist of multiple objects can be modeled as a whole or by focusing on each of their component objects individually. We selected to use the first approach to maintain a comprehensive view of aggregate objects. Thus, in the high level design phase, components of aggregate objects are modeled as a whole by specifying their characteristics under the architecture of aggregate object. In the low level phase of the design, roles of the component objects are revealed clearly in the fine grained details of design specifications of aggregate objects.

Designers should analyze the interface description and use the VRID model to specify characteristics of each virtual object, as described below.

HL3.1: Graphics

In this step, the goal is to specify a high-level description of graphics needs of virtual objects. Designers should describe what kinds of graphical representations are needed for each object, and its parts, if any. Since representing objects graphically is a creative task,
this step aims to provide flexibility to graphical designers by focusing only on general, high-level descriptions of graphical needs.

In our example, we should describe graphics needs of the patient body, organs, nerves, vessels, and muscles in enough detail for graphics designers to understand the context of the graphical modeling needs. We should mention that we need graphical model of an adult human body, which lies on its back on the operation table. Gender is arbitrary. Abdominal part of the body should be open, and show the organs in the area, and the nerves, vessels, and muscles that weave the organs. Boundaries of organs, nerves, vessels, and muscles must be distinguishable when highlighted.

HL3.2: Behaviors

The goals of this step are to identify behaviors exhibited by objects; classify them into simple physical, simple magical, or composite behavior categories; and to describe them in enough detail for designers to visualize the behaviors. This step involves the following activities: a) identify behaviors from the description; b) classify the behaviors into simple and composite categories; c) classify simple behaviors into physical and magical behavior categories; and d) for composite behaviors, specify sequences in which simple behaviors are to be combined for producing the composite behaviors.

In our example, behaviors exhibited by the patient body are: 1) bleeding; 2) highlighting nerves, vessels, and muscles with unique colors; and 3) showing the status of operation. Bleeding can be specified as a simple behavior or as a composite behavior obtained by combining simple behaviors of increasing the amount, color intensity, and viscosity of blood. This design decision should be based on reusability considerations. If the simple
behaviors can be reused in generating different behaviors of blood, it may be worth to specify bleeding as a composite behavior. Highlighting nerves, vessels, and muscles is a simple behavior whereas showing the status of operation is a composite behavior that consists of bleeding and highlighting behaviors. Bleeding is a physical behavior because there is nothing magic about it and it can be observed in real world. Highlighting nerves, vessels, and muscles with unique colors is a magical behavior because it has no counterpart in the real world.

The next task is to identify the combination and sequence in which the composite behavior of showing the status of operation is generated. The description indicates that bleeding and highlighting behaviors should be superimposed and exhibited simultaneously.

**HL3.3: Interactions**

The goal in this step is to specify where inputs of interface objects come from and how they change object behaviors. This step involves: a) identifying inputs to objects; b) identifying behavioral changes caused by inputs; and c) deciding on what requests should be sent to which behavioral components to bring about the changes.

*In our example, inputs into the patient body are the 3D coordinates of the needle. The interaction component should be able to process this data and interpret its meaning to decide whether the surgeon is prodding or puncturing (since this is a hypothetical example, we do not specify in detail how coordinates are to be processed and interpreted). If the surgeon is prodding, the implication for the behavior of the patient body is to show the status of operation. The interaction component should communicate
with the composite behavior component to initiate the "showing the status of operation" behavior. If the surgeon is puncturing, the implication for the behavior of the patient body is bleeding. The interaction component should communicate with the composite behavior component to initiate the bleeding behavior.

HL3.4: Internal communications (mediator)

In this step, the goal is to specify control and coordination needs for internal communications among the components of objects in order to avoid potential conflicts in object behavior. This involves: a) examining all behaviors of the object; b) identifying potentially conflicting behaviors; c) identifying communications requests that may cause the potential conflicts; and d) deciding how to prioritize, sequence, hold or deny the communications requests to avoid the potential conflicts.

In our example, behaviors of the patient body are: 1) bleeding; 2) highlighting nerves, vessels, and muscles with unique colors; and 3) showing status of operation. There are no potential conflicts among these behaviors because (3) is already defined as a legitimate behavior that is made up of a combination of (1) and (2). Hence, in this example, there is no need for a particular sequencing and prioritization of communications requests.

HL3.5: External communications

In this step, the goal is to specify control and coordination needs for external communications of the objects. This involves a) identifying communication inflows into the object, and their sources; b) communications outflows from the object, and their
destinations; and c) describing time and buffering semantics of external communications of the object.

In our example, communication inflows into the patient body are 3D coordinates coming from the needle. There are no communications outflows. Although it is not specified in the description, for illustrative purposes, we assume that a communication between the needle and patient body starts when the needle initiates an operation (e.g., prodding, puncturing), and the patient body is ready to display the associated behavior with that operation. If the patient body is busy, the needle will wait for a specified amount of time. If the patient body is still not ready after a certain amount of time, the needle will abort the communication request.

LOW-LEVEL (LL) DESIGN

Output of the high-level design becomes the input to the low-level design, which repeats the five modeling steps at a lower level abstraction to generate fine-grained details of the high-level design specifications:

- Step1 - (LL1) Graphics
- Step2 – (LL2) Behaviors
- Step3 – (LL3) Interactions
- Step4 – (LL4) Internal communications (mediator)
- Step5 – (LL5) External communications
The goal of the low level design phase is to provide low level, implementation oriented details of the high level design specifications using well known languages, techniques, and frameworks that we discussed in chapter 2. We provide a synthesis of existing languages, techniques, and frameworks that can be used in the low level phase of the VRID methodology based on the multi-component object architecture of the VRID model. Behavioral languages facilitate specification of low level details of behaviors at behavioral specification step of the low level design. The behavioral library can be used to organize magical and physical behaviors. Frameworks and models that were developed for behavioral characteristics of VR interfaces offer different ways to represent low level details of behaviors using implementation oriented terminology. For example, Cremer’s framework proposes to use hierarchical, communicating and concurrent state machines to represent behaviors of objects in real time virtual environments. These framework and models also provide communication mechanisms that can be used in internal and external communications step of the low level design. Similarly, previous work on interaction techniques helps to specify user interactions in the low level design by clarifying how user interacts with interface. Frameworks and models organize interaction techniques and provide design notations to represent low level details of interactions. Interaction languages facilitate specification of low level details of interaction. Table 4.1 provides a list of existing languages, techniques and frameworks that can be used in the low level design phase of the VRID methodology.

**LL1: Graphics**

Low-level design of graphics aims to associate graphical models and behaviors of objects. The outcome of this step should enable graphical designer to understand how
<table>
<thead>
<tr>
<th>Low level design steps</th>
<th>Previous work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Behaviors</td>
<td>Languages</td>
</tr>
<tr>
<td></td>
<td>• Green and Halliday 1996</td>
</tr>
<tr>
<td></td>
<td>• Steed and Slater 1996</td>
</tr>
<tr>
<td></td>
<td>Behavioral library:</td>
</tr>
<tr>
<td></td>
<td>• Stansfield, Shawver et al. 1995</td>
</tr>
<tr>
<td></td>
<td>Frameworks and models</td>
</tr>
<tr>
<td></td>
<td>• Cremer, Kearney et al. 1995</td>
</tr>
<tr>
<td></td>
<td>• Blumberg and Galyean 1995</td>
</tr>
<tr>
<td></td>
<td>• Tu and Terzopoulos 1994</td>
</tr>
<tr>
<td></td>
<td>• Gobbetti and Balaguer 1993</td>
</tr>
<tr>
<td>Interactions</td>
<td>Languages</td>
</tr>
<tr>
<td></td>
<td>• Jacob, Deligiannidis et al. 1999</td>
</tr>
<tr>
<td></td>
<td>• Smith and Duke 1999</td>
</tr>
<tr>
<td></td>
<td>Interaction techniques</td>
</tr>
<tr>
<td></td>
<td>• Bowman and Hodges 1997</td>
</tr>
<tr>
<td></td>
<td>• Liang and Green 1993</td>
</tr>
<tr>
<td></td>
<td>• Poupyrev,Billinghurst et al. 1996</td>
</tr>
<tr>
<td></td>
<td>• Stoakley, Conway et al. 1995</td>
</tr>
<tr>
<td></td>
<td>• Tanriverdi and Jacob 2000</td>
</tr>
<tr>
<td></td>
<td>• Wloka and Greenfield 1995</td>
</tr>
<tr>
<td></td>
<td>Frameworks and models</td>
</tr>
<tr>
<td></td>
<td>• Kessler 1999</td>
</tr>
<tr>
<td></td>
<td>• Lewis, Koved et al. 1991</td>
</tr>
<tr>
<td></td>
<td>• Gobbetti and Balaguer 1993</td>
</tr>
<tr>
<td></td>
<td>• Bowman 1999</td>
</tr>
<tr>
<td>Internal and external communications</td>
<td>Frameworks and models</td>
</tr>
<tr>
<td></td>
<td>• Cremer, Kearney et al. 1995</td>
</tr>
<tr>
<td></td>
<td>• Blumberg and Galyean 1995</td>
</tr>
<tr>
<td></td>
<td>• Tu and Terzopoulos 1994</td>
</tr>
<tr>
<td></td>
<td>• Gobbetti and Balaguer 1993</td>
</tr>
</tbody>
</table>

Table 4.1 The list of languages, techniques and frameworks that can be used in the low level design phase of the VRID methodology

Object behaviors can be animated. This step involves matching the graphical models specified in HL3.1 with behaviors specified in HL3.2.

*In our example, graphical models were specified for patient body and its parts (organs, nerves, vessels, and muscles in the abdominal area) in HL3.1. Associated behaviors specified in HL3.2 were bleeding, highlighting, and showing the status of operation.*
Using the description, we need to associate graphical models of organs with all three behaviors; and the graphical models of nerves, vessels, and muscles with the bleeding and highlighting behaviors.

**LL2: Behaviors**

In low-level design of behaviors, the goal is to formalize fine-grained procedural details of behaviors that have been specified in HL3.2. Formal representation of behaviors requires use of constructs of a selected design language. In our example, we use PMIW, the user interface description language that we had originally developed for specifying interactions (Jacob, Deligiannidis et al. 1999), but is also suitable for specifying behaviors. PMIW representation requires: a) identification of discrete and continuous components of behaviors; b) use of data flow diagrams to represent continuous behaviors; and c) use of statecharts (Harel and Naamad 1996) and state transition diagrams to represent discrete behaviors. For illustration purposes, we depict in Figure 4.3 formal representation of continuous and discrete parts of the bleeding behavior using PMIW.

**LL3: Interactions**

Like low-level design of behaviors, low-level design of interactions aims to formalize fine-grained aspects of the interactions that have been specified in HL3.3. Formal representation of interactions also requires selection of a design language. PMIW is well suited for this purpose, although designers may choose any other suitable design language. Activities outlined for formal representation of behaviors are repeated in this step, this time for representing interactions. For illustration purposes, we depict in Figure 4.4 a formal representation of the puncturing interaction using PMIW.
(a) Continuous parts of bleeding behavior

(b) Discrete parts of bleeding behavior

Figure 4.3 PMIW representation of bleeding behavior

Figure 4.4 PMIW representation of puncturing interaction

LL4: Internal communications (mediator)

In low-level design of internal communications, the goal is to specify scheduling mechanisms for managing the communication requests identified in HL3.4 as giving rise to conflicting object behaviors. The technique proposed by Reynolds (1987) to resolve
conflicts in competing goals of flocking behaviors can be used in specifying scheduling mechanisms. Reynolds' technique resolves conflicts by priority ordering of competing behaviors. Designers can also select from scheduling techniques commonly used in operating systems. Different techniques make different tradeoffs among minimum response time, maximum throughput, and maximum efficiency. Since most VR systems promise realistic, immersive environments, designers should pay special attention to techniques with better response times.

*In the virtual surgery example, we do not specify any particular scheduling technique since HL3.4 revealed that there are no potentially conflicting object behaviors.*

**LL5: External communications**

Low-level design of external communications aims to specify the message passing mechanisms that control and coordinate external communications of the objects. Designers can select from the synchronous, asynchronous, timeout, and bulking message passing mechanisms discussed by Booch (1991).

*In our example, the communication needs between the patient body and the biopsy needle, which had been specified in HL3.5, can be modeled with the timeout mechanism.*

This step concludes the first pass of the methodology phases. Designers should iterate between the modeling steps of the high and low level design phases, and refine the specifications until they are convinced that conceptually sound and implementable specifications are produced.
CHAPTER 5: GORILLA EXHIBIT EXAMPLE

In this chapter, we will apply the VRID model and methodology to an example and compare the VRID with previously built model and methodologies for conventional interfaces. The example is adapted from the interface of “Virtual Reality Gorilla Exhibit” system that is built by Allison et al at Georgia Institute of Technology (1997). “Virtual Reality Gorilla Exhibit” enables users to enter into one of the gorilla habitats as an adolescent gorilla at Zoo Atlanta in Atlanta, Georgia, and to interact as part of a gorilla family unit. Since the system contains most of the distinctive characteristics of VR interfaces that we discussed in chapter 1, it is a good example to illustrate the benefits of the VRID model and methodology. Furthermore, it is one of the well-known projects in VR field. The description of our example is adapted from the version of the system that is used in the preliminary user testing and is shown at Figure 5.1.

The Virtual Reality Gorilla Exhibit lets the user assume the persona of an adolescent gorilla, enter into a gorilla habitat, and interact as part of a gorilla family unit. The user can walk around the virtual environment and adjust the speed of walking using the handheld joystick. There are two gorillas in the habitat, an adult and an infant gorilla. Adult gorilla sits on the ground. Infant gorilla walks around the habitat. If the infant gorilla moves out of its play zone, adult gorilla calls the infant by moving towards it and slapping the ground. If the user approaches to the adult gorilla in threatening manner by walking quickly towards gorilla and staring continuously, the adult gorilla becomes annoyed and displays charging and chest-beating gestures. If the annoying situation continues until a specific time limit, the adult gorilla shows the user a banner that says “out of habitat” and the user finds himself/herself at the entrance point.

Figure 5.1 Description of Gorilla Exhibit
APPLYING VRID METHODOLOGY TO THE GORILLA EXHIBIT:

Throughout the example, designer will refer to the interface designer who uses the VRID model and methodology to design the system and user will refer to the end user who will use the system. As we emphasized before, designer should iterate between the steps of the methodology. For simplicity, we provide the version of the specifications that are refined after multiple iterations.

High Level Design Phase:

HL1. Identifying data elements

<table>
<thead>
<tr>
<th>Identify data inflows coming into the VR interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>joystick input; source: joystick</td>
</tr>
<tr>
<td>head coordinates; source: magnetic tracker</td>
</tr>
<tr>
<td>eye coordinates; source: eye tracker</td>
</tr>
</tbody>
</table>

Figure 5.2 Identification of data elements

In this step, we determine incoming data to the VR interface with their sources. The identification of data elements may not be complete in the first pass. For example, joystick inputs can be easily identified from the system description. However, head and eye coordinates may become more obvious after iterating between the steps of the methodology.

Identifying data elements specifically in the design helps to determine how a VR system communicates with its external environment. When specifying communications in the
later steps of the methodology, designers can refer to data elements without further
details. Design models and methodologies used in conventional interfaces specify
incoming data mostly in the stage where user commands and actions are described. Since
incoming data is usually restricted to keyboard and mouse inputs, this approach works
well for conventional interfaces. VR interfaces however, offer more diverse and
sophisticated ways of data inflows from a variety of sources. In this example, for
instance, head and eye coordinates come from the devices that track head and eye
movements of the user. Therefore, devoting specific attention to specify data elements is
essential in VR interfaces.

**HL2. Identifying objects**

<table>
<thead>
<tr>
<th>a)</th>
<th>identify potential objects mentioned in the interface description;</th>
<th>adolescent gorilla, adult gorilla, infant gorilla, joystick</th>
</tr>
</thead>
<tbody>
<tr>
<td>b)</td>
<td>decide on legitimate objects;</td>
<td>adolescent gorilla, adult gorilla, infant gorilla, joystick</td>
</tr>
<tr>
<td></td>
<td>user operates the adolescent gorilla</td>
<td></td>
</tr>
<tr>
<td>c)</td>
<td>distinguish between virtual and physical objects.</td>
<td>adolescent gorilla, adult gorilla and infant gorilla are virtual</td>
</tr>
<tr>
<td></td>
<td>joystick is physical</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 5.3 Identification of objects**

In this step, we identify objects in the gorilla exhibit and distinguish them based on
virtual or physical existence in the interface. Gorillas are virtual objects in the interface
since they need to be modeled and generated by computer. We also point out a difference
between gorillas such that adolescent gorilla is different from other virtual objects since it
represents the user in the interface. When a representative object for user i.e. avatar does
not exist, set of data elements coming from interaction devices that are used by the user,
represents the user in the interface. We specified the other object, joystick, as a physical object of the interface since it does not need modeling in the interface. Instead, inputs coming from joystick are used as data elements in the interface as it is described in the previous step.

Similar to design models and methodologies for conventional interfaces, the VRID model and methodology identifies entities that have well defined roles and identities in interfaces before proceeding to design details. Unlike other model and methodologies, the VRID emphasizes existence of both virtual and physical objects in VR interfaces. Proper alignment of virtual and real objects with respect to each other, also known as registration, is a challenging problem, especially in augmented reality interfaces (Azuma 1997). Thus, identifying both virtual and real objects and specifying their roles in the design enable designers to focus on registration problem in the early phases of the design.

HL3. Modeling objects

HL3.1 Graphics

describe what kinds of graphical representations are needed for each object, and its parts, if any.

Graphical models of adolescent gorilla, adult gorilla and infant gorilla are required. The adult gorilla should be 5” tall. The adolescent gorilla should be 2/3 of size of the adult gorilla. The infant gorilla should be 1/2 of size of the adolescent gorilla. The hair on the back of the adult gorilla is gray, while hairs of other gorillas are black.

Figure 5.4 High level graphical specifications

In this step, we try to give graphical designers enough information about distinctive graphical characteristics of each gorilla. Thus our aim is to inform graphical designers
about specific graphical requirements of the interface and to provide flexibility to them for the rest of the graphical details.

Since most of the VR interfaces are generated using 3D graphics, they are visually more demanding to build than conventional interfaces. Especially in interfaces where objects exhibit behaviors, it is important to match the visual appearance with object dynamism to provide immersive and realistic environments. Thus, by including graphical specifications to the VRID model and methodology, we aim to provide a good communication mechanism between graphical design and the rest of the design of the interface. None of the previous design models and methodologies for conventional interfaces includes graphics as a component in their structures, except the OAI model. Because of its focus on GUI style interfaces, the OAI model emphasizes the metaphoric representation of abstract GUI objects. This approach is not sufficient for VR interfaces that contain objects such as real world like creatures. These objects are graphically complex and each of their graphical components may have specific requirements that need to be addressed specifically in the interface specifications. For instance, the gorillas in our example have 3D graphical models. Each component of the graphical model of a gorilla represents a different part of gorilla’s body that requires different behavior and interaction specifications in the interface.

**HL3.2 Behaviors**

<table>
<thead>
<tr>
<th>a) identify behaviors from the description;</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>adolescent gorilla:</em> walking around, approaching to the adult gorilla in threatening manner, walking quickly towards gorilla, staring continuously to</td>
</tr>
</tbody>
</table>
gorilla, returning to entrance

adult gorilla: sitting on the ground, calling infant, moving towards infant, slapping ground, becoming annoyed, displaying charging and chest beating gestures, showing a banner to user

infant gorilla: walking around

b) classify the behaviors into simple and composite categories:

adolescent gorilla:
walking around: composite
approaching to the adult gorilla in threatening manner: composite
walking quickly towards the adult gorilla: composite
staring continuously to gorilla: simple
returning to entrance: simple

adult gorilla:
sitting on the ground: simple
calling the infant: composite
moving towards the infant: composite
slapping ground: simple
becoming annoyed: composite
displaying charging gesture: composite
displaying chest beating gesture: simple
showing the banner to user: composite

infant gorilla:
walking around: composite

c) for composite behaviors, specify sequences in which simple behaviors are to be combined for producing the composite behaviors.

adolescent gorilla:

walking around: 1. moving feet in normal speed 2. positioning body using the head direction of the user (simultaneously)

approaching to the adult gorilla in threatening manner: 1. walking quickly towards gorilla 2. staring continuously (simultaneously)

walking quickly towards gorilla: 1. moving feet in high speed 2. positioning body using the head direction of the user, which is towards the adult gorilla (simultaneously)

adult gorilla:
calling the infant: 1. moving towards the infant 2. slapping ground. (simultaneously)

moving towards the infant: 1. moving feet in high speed 2. positioning body towards the infant gorilla (simultaneously)

becoming annoyed: 1. displaying charging gestures 2. displaying chest beating gestures (simultaneously)

displaying charging gestures: 1. rushing forward 2. attacking position (serially)

showing a banner to user: 1. hold the emerged banner with front and rear feet on the standing position 2. lift the banner up (serially)

infant gorilla:
walking around: 1. moving feet in high speed 2. positioning body towards random direction (simultaneously)

d) classify simple behaviors into physical and magical behavior categories;

adolescent gorilla:
moving feet: physical
positioning body: physical
staring continuously to gorilla: physical
returning to entrance: magical (the virtual world that user is experiencing suddenly changes and user finds himself/herself at the entrance point)

adult gorilla:
sitting on the ground: physical
moving feet: physical
positioning body: physical
slapping ground: physical
rushing forward: physical
attacking position: physical
displaying chest beating gesture: physical
hold the emerged banner with front and rear foot on the standing position: magical
lift the banner up: magical

infant gorilla:
moving feet: physical
positioning body: physical

Figure 5.5 High level behavioral specifications
In this step, we identify behaviors of each gorilla and analyze them to have a better understanding about their design requirements. Some behaviors like “walking around”, are initiated by gorillas autonomously, whereas other behaviors like “calling the infant” or “becoming annoyed” results from communications with other gorillas in the exhibit.

Classifying behaviors as simple or composite is based on the decision of the designer. However, the VRID model and methodology encourage specifying a behavior as composite if it helps to decrease the complexity or increase the reusability of the behavior. For example, “displaying charging gestures” behavior of the adult gorilla may not be simple enough to understand its exact meaning. However, it may help to specify it as a composite behavior and to divide it into simpler components as we did above. To increase the reusability of behaviors, for example, we classify “walking” behaviors of gorillas as composite behaviors since it enables us to use the components in different ways to specify various types of “walking behaviors”.

As we described in the model, further classification of simple behaviors as physical or magical help designers to generate a library of simple physical and magical behaviors. These libraries will be helpful to organize behaviors and to simplify their reusability for similar situations. For example, if physical behaviors exhibited by gorillas are stored in the library, they can be used if another gorilla needs to be added to the interface. Furthermore, physical and magical classification enable designers to assess the level of specifications required for simple behaviors. Physical behaviors have counterparts that can be commonly seen in the real world. Therefore, designers may omit to describe all
the details of physical behaviors, and focus on the details that require particular attention in the design. Magical behaviors, on the other hand, are hardly or impossible to be seen in the real world. Therefore, designers may need to describe these behaviors in detail to avoid any errors caused by misunderstandings in the later stages of the development. For example, “sitting on the ground” or “moving feet” behaviors of the adult gorilla do not require detailed descriptions, since it is easy to associate them with their real world counterparts. “Lift the banner up” behavior of the gorilla, on the other hand, may require further descriptions, as it is hardly seen in the real world.

Among the previously developed design model and methodologies that we discussed before, object oriented design model and methodologies are the only ones that support object dynamism. However, they provide guidance in low level specification details. Lack of high level support for object dynamism makes it harder to understand and to specify complex behaviors such as “approaching to the adult gorilla in threatening manner” behavior of the adolescent gorilla.

**HL3.3 Interactions**

<table>
<thead>
<tr>
<th>a)</th>
<th>identify interaction requests to objects;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><em>adolescent gorilla.</em></td>
</tr>
<tr>
<td></td>
<td>joystick input (walk input, speed input), head coordinates, eye coordinates</td>
</tr>
<tr>
<td></td>
<td><em>adult gorilla:</em></td>
</tr>
<tr>
<td></td>
<td>gaze direction input</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>b)</th>
<th>identify behavioral changes caused by these requests and which behavioral components will be notified for these changes;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><em>adolescent gorilla.</em></td>
</tr>
</tbody>
</table>
Figure 5.6 High level interaction specifications

As we emphasized in the first chapter, VR interfaces support both explicit and implicit commands. In this example, user interacts with the interface by explicit commands to walk and to change speed of walking using joystick. Implicit commands of the user come from his/her head and eye coordinates. Unlike the explicit commands, interaction component receives inputs for the implicit commands continuously to determine the meaning of user actions. Interaction component of adolescent gorilla, for example, receives head and eye coordinates of the user continuously to determine the user's head position and where the user looks at and adjusts its position.
Models and methodologies for conventional interfaces support specifications of explicit style interactions in the form of user commands to the interface. The VRID model and methodology on the other hand, receive user requests coming from various inputs and interprets their meaning to decide resulting interactions. This approach facilitates to determine both explicit and implicit style interactions easily. Furthermore, unlike other model and methodologies, the VRID help designers to specify how interactions and behaviors are related by emphasizing associations between interactions and possible resulting behaviors. For example, interaction component of the adult gorilla interprets implicit interaction that results from gaze direction input of the user and decides if the user is just looking or staring at the gorilla. If the user is staring at the gorilla, this causes a change in the behavior of the gorilla to “becoming annoyed”. Conventional models and methodologies would not help the designer to make this association.

**HL3.4 Internal communications (mediator)**

<table>
<thead>
<tr>
<th>Examine all communication requests and behavioral changes that are caused by these requests.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>adolescent gorilla:</strong></td>
</tr>
<tr>
<td>“out of habitat”; behavioral change: return to the entrance</td>
</tr>
<tr>
<td><strong>adult gorilla:</strong></td>
</tr>
<tr>
<td>“user is out”; behavioral change: show banner</td>
</tr>
<tr>
<td>“infant is out of zone”; behavioral change: call infant</td>
</tr>
<tr>
<td>“stare timeout”; behavioral change: becoming annoyed</td>
</tr>
<tr>
<td>a) Identify communications requests that may cause the potential conflicts;</td>
</tr>
<tr>
<td>There are no potential conflicts among the behaviors of the adolescent gorilla. However, “out of habitat” request must be processed immediately, even if other</td>
</tr>
</tbody>
</table>
communication requests already exist for the adolescent gorilla. For the adult gorilla, “stare timeout” and “infant is out of zone” can be conflicting. If these requests happen at the same time, the adult gorilla cannot perform the associated behaviors simultaneously. If the gorilla receives “user is out” request with other requests, it must process “user is out” immediately.

b) decide how to prioritize, sequence, hold or deny the communications requests to avoid the potential conflicts

“out of gorilla” must have the highest priority among the communication requests for the adolescent gorilla. Similarly, for the adult gorilla, “user is out” must have the highest priority. We assume that infant’s safety has higher priority than initiation of “becoming annoyed” behavior for the adult gorilla. In this case mediator of the adult gorilla gives higher priority to “infant is out of zone” request than “stare timeout” request.

| Figure 5.7 High level internal communications specifications |

Despite its limited scope, this example indicates various communications between the gorillas. This step helps us to organize processing of these communications internally in each gorilla. We specify how communication requests affect behaviors of each gorilla. In addition, we provide solutions to potential conflicts that may emerge among different communication requests.

As we described in the first chapter, complex communication patterns among objects are one of the distinctive characteristics of VR interfaces. The VRID model and methodology help to decompose this complexity by separating external and internal communication specifications for each object. External communication specifications, which are the focus of the next step in the methodology, focus on handling incoming and outgoing communication requests for objects. Internal communication specifications provide guidance in describing internal processing in an object as a result of incoming communication requests. Furthermore, it helps designers to provide solutions for possible
conflicts in the process of communication requests before they become more difficult and costly in the implementation. Lack of support for communication among objects is one of the major drawbacks of using previously developed design models and methodologies in VR interfaces. Among the models and methodologies that we discussed previously, Foley's model and object oriented design model and methodologies include object communications in their architecture. Foley's model requires specification of relationships between objects. However, the model does not provide details on how to specify these relationships. On the other hand, the approach of object oriented design model and methodologies is that they provide support to specify communications among objects in lower level details and they do not address problems that require higher level view such as solutions to possible conflicts among different communication requests.

HL3.5 External communications

a) identify communication inflows into the object, and their sources;

  adolescent gorilla:
  joystick input; source: joystick;
  eye coordinates; source: eye tracker
  head coordinates; source: magnetic tracker
  "out of habitat"; source: adult gorilla

  adult gorilla:
  "user is out"; source: application
  "infant is out of zone"; source: application
  "stare timeout"; source: application
  gaze direction; source: adolescent gorilla

b) identify communication outflows from the object to application or external entities
infant gorilla:
3D coordinates of its position; destination: application (decides if the infant is out of zone)

c) describe time and buffering semantics of external communications of the object.
adolescent gorilla.
Joystick input, head and eye coordinates input are sent to the adolescent gorilla regardless of whether the gorilla is ready to receive the message. Communication between the adolescent gorilla and the adult gorilla for “out of habitat” starts when the adult gorilla shows the banner to the adolescent gorilla.

adult gorilla:
Application sends “user is out” request to the adult gorilla if the timeout for annoying adult gorilla is reached and when the adult gorilla is ready to receive the message. Similarly, application sends “infant is out of zone” if the infant gorilla is out of its play zone and when the adult gorilla is ready to receive the message. The last communication request from application, “stare timeout” comes from application when the timeout for staring is reached and application will wait for a specific amount of time for the adult gorilla to be ready.

infant gorilla:
Infant gorilla sends its 3D coordinates to the application continuously

Figure 5.8 High level external communications specifications

In the previous step of the methodology, we specify how communications are organized and processed internally in each gorilla. In this step, we describe communication of each gorilla with other gorillas or with the application using communication component. For simplicity, if communication is between two objects in the interface, we specify the communication from recipient’s perspective.

As we mentioned above, external communication focuses on communication of objects with their outside environment. Especially in dynamic virtual environments where objects communicate with each other as in our gorilla example, it is important to manage and organize the message traffic.
This step concludes the high-level design phase of the methodology. The high-level design phase guided us to determine the design specifications of the interface in broader terms using the language of the interface domain i.e. gorilla habitat. This made it easier to clarify the requirements that are described in the interface description. Furthermore, the high-level view helped us to identify relationships between components of multi-component object architectures of gorillas.

**Low Level Design Phase:**

**LL1. Graphics**

match the graphical models specified in HL3.1 with behaviors specified in HL3.2.

*adolescent gorilla.*

Moving feet is associated with the graphical models of feet of the adolescent gorilla. The rear foot should move quickly and the front foot should support the movements or the rear foot.

Positioning body is associated with the graphical models of trunk and head of the gorilla's body.

Staring continuously is associated with the graphical model of the head and eye of adolescent gorilla.

returning the entrance is associated with the graphical model of the environment that surrounds the adolescent gorilla. The environment changes from the habitat to the entrance region.

*adult gorilla:*

Sitting on the ground is associated with the graphical models of the feet and trunk. Moving feet and positioning body are the same as specifications for the adolescent gorilla.

Slapping the ground is associated with the graphical models of the front feet.

Rushing forward is associated with the graphical models of the feet.
Attacking position is associated with the graphical models of the trunk, head and front feet.
Displaying chest beating gesture is associated with the graphical models of the trunk and front feet.
Hold the emerged banner with front and rear feet on the standing position is associated with the graphical models of the feet and trunk.
Lift the banner up is associated with the graphical models of the front feet.

**infant gorilla:**
Moving feet and positioning body are the same as specifications for the adolescent gorilla.

**Figure 5.9 Low level graphical specifications**

In this step, we describe association of behaviors with parts of graphical models of objects. Our goal is to clarify our expectations from the graphical design of the gorillas so that there will be no mismatch between intended behaviors and visual appearances for each gorilla. We keep our specifications short and simple, but they can be as detailed as necessary.

This step contributes to the integration of the graphical part of the interface with the rest of the interface design from a lower level of abstraction. It provides guidance to designers to build a direct relationship between graphics and behaviors so that animations generated by graphical designers and intended behaviors specified by interface designers would be compatible.

**LL2. Behaviors**

formalize fine-grained procedural details of behaviors that have been specified in HL3.2 using constructs of a selected design language.
We use PMIW as a design language to specify low-level design of behaviors (Jacob, Deligiannidis et al. 1999). Some behaviors have either continuous or discrete parts, and some behaviors have both. Continuous parts of the behaviors are represented with data flow diagrams, whereas discrete parts are represented with state transition diagrams. For example, “approaching in threatening manner” behavior of the adolescent gorilla starts if “threaten” variable becomes active and continues until the variable becomes inactive. This variable is controlled by the interaction component of the adolescent gorilla, which will be specified in the next step. “Returning to the entrance” behavior of the adolescent gorilla, however, has only discrete components, since it happens as soon as “out of habitat” variable becomes active.

*adolescent gorilla.*

(a) Continuous parts of approaching in threatening manner behavior

(b) Discrete parts of approaching in threatening manner behavior
(a) Continuous parts of walking behavior

(b) Discrete parts of walking behavior

OUTOFHABITATION

start

Return to the entrance

Returning to the entrance behavior

adult gorilla.

(a) Continuous parts of calling the infant behavior

(b) Discrete parts of calling the infant behavior
(a) Continuous parts of moving towards infant behavior

(b) Discrete parts of moving towards infant behavior

(a) Continuous parts of showing banner behavior

(b) Discrete parts of showing banner behavior

(a) Continuous parts of becoming annoyed behavior

(b) Discrete parts of becoming annoyed behavior
Flexibility of the VRID on the selection of the language to specify behaviors helps designers to adapt a language that fits best to their needs to the design process. This is particularly important for VR interfaces since objects may exhibit a wide range of behaviors. In this example, for instance, we chose PMIW since it facilitates specification of both continuous and discrete types of behaviors.

**LL3. Interactions**

formalize fine-grained aspects of interactions that have been specified in HL3.3 using constructs of a selected design language.

Similar to the low level design of the behaviors, we select PMIW to specify low level design of interactions. The adolescent gorilla receives head and eye coordinates of the user continuously to determine its direction in the gorilla exhibit. Therefore, components of the interaction that processes head and eye coordinates do not have discrete parts.
Similarly, the interaction component of the adult gorilla, which processes gaze direction of the user, has only continuous parts. Interactions for walking and changing speed of the adult gorilla, however, are only enabled while the user presses the corresponding joystick buttons down. Since the interaction is temporary and ends when the user presses joystick buttons up, components of the interaction that processes walking and speed inputs have both discrete and continuous parts.

adolescent gorilla:

![Diagram]

a) Continuous part of interaction component details of the adolescent gorilla

![Diagram]

b) Discrete part of interaction component details of the adolescent gorilla

adult gorilla:

![Diagram]

Processing gaze

Interaction component details of the adult gorilla

Figure 5.11 Low level interaction specifications with PMIW
Like in behavioral specification, the VRID provides flexibility in selection of an appropriate interaction language for the needs of the design. This is particularly important for VR interfaces, since they offer potential for both explicit and implicit style interactions. It is hard to see such flexibility in the models and methodologies for conventional interfaces.

**LL4. Internal communications (mediator)**

Specify scheduling mechanisms for managing the communication requests identified in HL3.4 as giving rise to conflicting object behaviors.

*To allocate the priorities that are specified in HL3.4, priority scheduling can be used for both adolescent and adult gorilla. For the adolescent gorilla, “out of habitat” has the highest priority, while priorities for the other requests can be the same. For the adult gorilla, highest priority is given to “user is out” request. Next higher priority is given to “infant is out of zone“. Finally, “stare timeout” is given the least priority.*

**Figure 5.12 Low level internal communications specifications**

In this step, we provide details on how to solve the potential conflicts in internal communications that are specified in the high level phase of the design. As in the previous steps of the low level phase, designers are free to choose appropriate scheduling mechanisms. In our example, we prefer to use priority scheduling mechanisms that are similar to the ones used in operating systems.

**LL5. External communications**

Specify the message passing mechanisms that control and coordinate external communications of the objects. Designers can select from the synchronous,
asynchronous, timeout, and bulking message passing mechanisms.

**adolescent gorilla.**  
Joystick inputs, head and eye coordinates: Asynchronous  
“out of habitat”: Synchronous

**adult gorilla:**  
“user is out”: Synchronous  
“infant is out of zone”: Synchronous  
“stare timeout”: Timeout

**infant gorilla:**  
position coordinates: Asynchronous

---

**Figure 5.13 Low level external communications specifications**

In this step, we determine message passing mechanisms using high level design specifications of time and buffering semantics for external communications. For example, we assume that application wait only for a specific amount of time for the adult gorilla to be ready to receive the “stare timeout” communication request. Therefore, we select timeout message passing mechanism. On the other hand, for the rest of the communication requests for the adult gorilla (“user is out” and “infant is out of zone”), we select synchronous message passing mechanism to make sure that application waits until the adult gorilla becomes ready to receive the request. Again, designers can choose an appropriate mechanism that fits best to their needs. At the end of this step, complexity of communications between the elements of the Gorilla Exhibit VR system are decomposed into distinct components which are easier to implement.

The completion of the low-level design phase takes the design of the “Gorilla Exhibit” VR interface one step forward to the implementation process. Languages and
mechanisms that we used throughout the steps of the low-level design phase enabled us to describe the design specifications with implementation-oriented terminology that can guide software developers in implementing the interface.

This step concludes a complete pass of the phases of the VRID methodology. In applying the VRID methodology, first we started with the English description of the VR system. Then throughout the steps of the VRID methodology, we decomposed the system into the pieces using components of the VRID model. At the end, each component of the interface provided a nicely encapsulated and conceptually distinct part of the interface, which will guide software developers through implementation process. The specifications can be refined further if necessary. However, we found the level of the detail of the specifications sufficient enough to demonstrate the applicability of the VRID methodology.

DISCUSSION

As we explained throughout the "Gorilla Exhibit", the VRID model and methodology guided us in designing the interface that exhibits all distinct characteristics of VR interfaces that we described in the first chapter. First, by distinguishing objects as virtual and physical, we clearly identified roles of each object in the interface and how they exchange information in the early stages of the design. For example, we specified joystick as a physical object in the high level design of the interface since no further modeling would be necessary for this object. Later in the low level design, we emphasized the role of the joystick in the design by specifying data elements that are
originated from it. Design models and methodologies developed for conventional interfaces do not support integration of virtual and physical objects in the design, which is an important factor especially for augmented reality interfaces.

Second, the VRID model and methodology enabled us to specify behaviors of gorillas easily in the example. Decomposing behaviors as composite and simple helped to decrease the complexity of behaviors and to reuse components of composite behaviors in specifying similar behaviors. Since active objects that can exhibit autonomous behaviors are rarely seen in conventional interfaces, models that are built for these interfaces do not support specifications of object behaviors.

Next, the VRID model and methodology provided us flexibility and support for specifying both implicit and explicit style interactions. To specify both implicit and explicit style interactions between user and gorillas in the habitat, we prefer to use the PMIW design language. However, designers can select any design language that fits to their needs. Since users interact with conventional interfaces using mostly explicit commands, models and methodologies for conventional interfaces do not provide such flexibility in specifying interactions.

Finally, the VRID model and methodology enabled us to specify both internal and external communications for the gorillas in the example. At the end of the design, we identified not only individual specifications for each gorilla, but also how they affect other gorillas in the interface. Active objects in VR interfaces, like gorillas in this
example, may interact with each other in various ways. Thus, for a complete and conceptually sound design, it is important to determine the communications that require attention of the VR interface and to provide appropriate response. Models and methodologies for conventional interfaces however, provide little or no support for communications among objects of an interface.

In addition to supporting distinct characteristics of VR interfaces, architecture of the VRID model and methodology provided further advantages in the design process. Multi component object architecture of the VRID model helped us to decompose design specifications of each object conceptually into distinct, easier to manage components; such decomposition is missing in current practice of VR interface designs.

Furthermore, the separation of high level and low level design phases of the VRID methodology provided a distinction between high level design considerations and low-level implementation details. Thus, in the high level phase, we analyzed the description of the system and identify the components of the interface and their characteristics without going into implementation specific details. This helped us to look at the entire system and examine its requirements from a higher level of abstraction. In the next phase, we elaborated these specifications and described implementation oriented, procedural details to guide software engineers in the development process.

Finally, the steps of the methodology encouraged us to consider various choices and tradeoffs during the design. We found it as an important factor in the success of the
design since it forced us to find the optimum solution from possible alternatives. For example, in specifying communications between gorillas, we considered different timing and buffering constraints to select appropriate communication mechanisms. As a result, communications between gorillas become more realistic and easier to implement later in the implementation process.
CHAPTER 6: ESTABLISHING THE VALIDITY OF THE VRID MODEL AND METHODOLOGY: AN EMPIRICAL STUDY

In this chapter, we present a user study that is conducted to test the validity, usability, and usefulness of the VRID model and methodology. This is an experimental study in which one group of designers designed a VR interface using the VRID model and methodology, and another group of designers, the control group, used the object oriented methodology for designing the same VR interface. We compared the two methodologies by assessing the performance of the designers, satisfaction of the designers with the respective methodologies that they used, and the quality of the resulting designs.

METHODO

We used a between-subjects experimental design. We formed two groups of designers. Both groups were asked to design the same VR interface.

Independent variable

In this experiment, the independent variable (manipulation) was the type of design model and methodology used by the two groups. One of the design groups used the VRID model and methodology whereas the other group used the OO model and methodology for designing the same VR interface.
Dependent variables

We compared the two models and methodologies by looking at their impact on the following dimensions:

1. **Performance of designers.** We measured performance of the designers by measuring the time they took to design the assigned VR interface. Our objective was to assess if the VRID model and methodology reduced the time to design VR interfaces.

2. **Designer satisfaction with the design model and methodology.** Since the VRID model and methodology seek to provide guidance to designers in the design task, our purpose was to understand if designers are satisfied with the model and the methodology. We measured designer satisfaction with the VRID and OO models and methodologies by asking them about:

   - Ease of learning the model and methodology
   - Ease of using the model and methodology
   - Degree of mental effort required during the design process
   - Degree of guidance provided by the design model in decomposing the overall design task into smaller tasks
   - Degree of guidance provided by the design model in making design tradeoffs
   - The extent to which the design model and methodology met the needs of the assigned VR interface.

3. **Quality of the resulting design.** Finally, we wanted to understand if the designs built with the VRID model and methodology are superior than the designs built with
alternative models and methodologies. Once a design is completed, it is handed over
to a software developer, who will in turn build on the design to begin coding the
interface. Therefore, we had a group of software developers rather than the original
designers, to assess the quality of the resulting designs. We asked the software
developers to assess the designs along the following dimensions:

- Is the design easy to understand?
- Does the design provide a high level view of the VR interface and give you an idea
  about the overall software coding task?
- Does the design provide sufficient detail for you to write the software code for the
  VR interface?
- Does the design provide a nice balance between high-level and low-level design
  specifications?
- Are the components specified in the design reusable in developing similar VR
  interfaces?
- Does the design do a good job in specifying the graphical representations of animals
  in the VR interface?
- Does the design do a good job in specifying the behaviors of the animals in the VR
  interface?
- Does the design do a good job in specifying the communications among the animals
  in the VR interface?
- Does the design do a good job in specifying user interactions with animals in the VR
  interface?
• Does the design do a good job in linking animal behaviors with graphical representations that are required for displaying those behaviors?

• Does the design reduce your mental workload in writing the software code for the VR interface?

• Overall, do you find this design successful?

Participants

Participants were 19 students from two computer science courses at the Electrical Engineering and Computer Science Department at Tufts University. Participation was voluntary. Participants received extra course credit in return for their participation. Participants assumed two roles in the study. In the first phase of the study, they served as designers. They designed a given VR interface using the design model and methodology that they were assigned to. In the second phase, they served as software developers who were responsible for taking a given VR interface design and beginning to write the code for the interface based on that design. None of the participants had any prior design experience with virtual reality software. Table 6.1 presents demographic characteristics of the participants.

<table>
<thead>
<tr>
<th>Demographics Questions</th>
<th>VRID</th>
<th>OO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>23.27</td>
<td>24</td>
</tr>
<tr>
<td>Number of computer languages that they know well</td>
<td>5.91</td>
<td>3.63</td>
</tr>
<tr>
<td>Number of software design projects that they did in their coursework</td>
<td>1.27</td>
<td>1.5</td>
</tr>
<tr>
<td>Number of software design projects that they did in the industry</td>
<td>1.55</td>
<td>0.13</td>
</tr>
</tbody>
</table>

Table 6.1 Characteristics of the participants

Note: M and SD represent mean and standard deviation respectively.
Use of student subjects rather than experienced professionals was appropriate for the purposes of this study. Our objective was to assess if the proposed VRID method and methodology are easy to learn and apply in designing VR interfaces. Experienced professionals are typically experts in using particular models, methodologies, and associated tools. It would be difficult for them to unlearn what they already know well and to learn the proposed VRID model and methodology. Similarly, the control group who use the OO model and methodology would introduce bias if we had used professionals who are already experts in using the OO model and methodology. Our student designers and developers had no prior experience with either the VRID or the OO model and methodologies. In fact, majority of them learned these models and methodologies for the first time in our training sessions. By using student designers and developers instead of professionals, we were able to minimize any potential experience and expertise bias in assessing the ease of learning, ease of use, and benefits of the VRID and OO models and methodologies.

Procedure

One group of designers was trained in the use of VRID model and methodology whereas the other group was trained in the use of object-oriented model and methodology. At the end of the training session, the designers received a relatively simple VR interface design task, which presented in Figure 6.1. The designers were given one week to do the design. They filled out two questionnaires. The first questionnaire was filled out before they started with the design. Its goal was to assess if the designers were trained sufficiently with the model and methodology that they were assigned to use during the design. On
average, both designer groups felt moderately knowledgeable about the use of the model and methodology in which they were trained. The second questionnaire was filled out after the completion of the design. It aimed to measure the time to complete the design, the satisfaction of the designers with the model and methodology they used, and the appropriateness and usefulness of the model and methodology in designing the given VR interface. In order to measure the time to complete the design, we provided designers with a timetable so that they could record the start and stop times every time they worked on the design task.

The first phase of the study was complete upon submission of the designs by the two groups. In the second phase, the objective was to have software developers to assess the resulting designs. We removed the names of the designers from the designs and redistributed the designs to the participants. We made sure that nobody received his or her own design or that of a close friend's. In this phase, participants served as software developers who are responsible for coding the given VR interface design. We asked software developers to review the design and assess various properties of the design by filling out a questionnaire. We also described a change to be made in the design and asked the software developers how they would modify the given design to incorporate the required change. We also asked them to estimate the time it would take them to make the required change. Our objective was to understand if the designs built with VRID model and methodology would be easier to modify than the designs built with OO model and methodology. We gave software developers one week to work on this phase of the study.
Description of the “Fun Animals” Virtual Reality System:
“Fun Animals” virtual reality system consists of four animals: a parrot, a bear, a bunny, and a spider. Initially, the parrot is periodically drinking water from a cup; the bear and the bunny are standing next to each other; and the spider is standing still on the ground. The user can interact with these animals and objects by using hand movements: a VR input device (e.g., a glove) tracks hand movements and sends coordinates of the hand to the VR system.
If the user changes the position of the cup, the parrot looks around and tries to find the cup. When the parrot finds the cup, it resumes drinking water periodically.
When the user touches either the bear or the bunny, the touched animal looks at the other animal and starts to expand gradually while the other animal starts to shrink gradually. If the user keeps touching the animal for more than a pre-specified duration, the animal explodes. After the explosion, both animals return to their initial sizes.
If the user moves his/her hand towards the spider, initially the spider stands still. If the distance between the hand and the spider becomes shorter than a predefined distance, the spider starts moving in the opposite direction of the user’s hand movement: i.e., if the hand moves to the right, the spider moves to the left. If the hand moves to the left, the spider moves to the right. If the spider comes closer to the parrot than a predefined distance, the parrot stops drinking water from the cup. If the user stops moving his/her hand, the spider stops moving.

Figure 6.1 VR interface design task

RESULTS
The first questionnaire filled out by the designers before the design did not indicate any significant differences between the level of knowledge and experience of the two design groups about the model and methodology in which they were trained. Six out of eight designers in the OO group indicated that they had no prior experience with object oriented design and that they learned it for the first time in our training session. Likewise, all of the designers in the VRID group learned the VRID model and methodology for the first time in our training session. Therefore, we are confident that our findings below are a function of the VRID and OO models and methodologies only. We can rule out
alternative explanations such as level of training, and prior experience and expertise of designers with the OO or VRID models and methodologies.

Performance results

Designers who used the VRID model and methodology in designing the VR interface performed better than the designers who used the OO model and methodology. The mean time to complete the design was 5.82 hours for the designers using the VRID model and methodology whereas it was 8.19 hours for the designers using the OO model and methodology. This result indicates that the VRID model and methodology reduced the time to design the VR interface by about 29%.

Designer satisfaction with the VRID and OO model and methodology

Designers in the VRID group were in general more satisfied with the design model and methodology that they used than the designers in the OO group. Table 6.2 presents the satisfaction-related questions and the responses of the designers. The responses indicate that the VRID model and methodology are easier to learn; provide better guidance in decomposing the overall design task into smaller, less complex, and conceptually distinct parts; better help designers to make design choices and tradeoffs; and better meet the needs of the VR interface. In terms of the ease of use of the model and methodology, the VRID group provided slightly lower ratings than the OO group, but the ratings were still satisfactory (i.e., 3.82/5 in VRID versus 3.88/5 in OO).
<table>
<thead>
<tr>
<th>Questions</th>
<th>VRID</th>
<th>OO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning to use the design model and methodology was easy for me.</td>
<td>3.91</td>
<td>3.63</td>
</tr>
<tr>
<td>Using the design model and methodology required a lot of mental effort.</td>
<td>2.82</td>
<td>3</td>
</tr>
<tr>
<td>The design model and methodology provided guidance in decomposing the</td>
<td>4.36</td>
<td>4.25</td>
</tr>
<tr>
<td>overall design task into smaller, less complex, and conceptually distinct parts.</td>
<td>1.03</td>
<td>0.71</td>
</tr>
<tr>
<td>The design model and methodology enabled me to understand various design choices and tradeoffs.</td>
<td>3.36</td>
<td>2.63</td>
</tr>
<tr>
<td>It was easy for me to use the design model and methodology in designing the assigned VR interface.</td>
<td>3.82</td>
<td>3.88</td>
</tr>
<tr>
<td>The design model and methodology met my needs in designing the assigned VR interface.</td>
<td>3.82</td>
<td>3.5</td>
</tr>
</tbody>
</table>

Table 6.2 Designers’ satisfaction with VRID and OO models and methodologies

Notes: All questions were posed using a 5-point Likert scale ranging from disagree (1) to agree (5). M and SD represent mean and standard deviation respectively.

We also captured qualitative comments from the designers by asking them the following open-ended question: “Please explain any major difficulties that you might have faced in designing this Virtual Reality interface.” The designers who used the OO design model and methodology reported difficulties in specifying interactions and communications among objects. The designers who used the VRID model and methodology, however, did not report any difficulties. These qualitative comments along with the quantitative scores confirm our assessment that existing design models and methodologies are too generic to guide the design of VR interfaces.

Evaluation of resulting designs

Satisfaction of software developers with the resulting designs. Rating of software developers, who were responsible for coding the VR interface based on the resulting designs, indicate that the designs built with the VRID model and methodology are higher in quality than the designs built with the OO design model and methodology. Table 6.3 presents the specific questions and the mean and standard deviations of the responses.
given by the software developers. As the table indicates, the designs built with VRID model and methodology were easier to understand. They provide a higher-level view of the overall coding task and sufficient detail for doing the actual coding. They provide a better balance between high-level and low-level design specifications. Reusability of design components is higher in the designs built with VRID model and methodology. Graphical, behavioral, interaction and communication aspects of the interface elements are better specified with the VRID model and methodology. Overall, software developers find the designs built with VRID model and methodology more successful than the designs built with the OO model and methodology.

<table>
<thead>
<tr>
<th>Questions</th>
<th>VRID</th>
<th></th>
<th>OO</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>This design is easy to understand.</td>
<td>3.82</td>
<td>1.17</td>
<td>3.88</td>
<td>0.84</td>
</tr>
<tr>
<td>This design provides a high-level view of the VR interface and gives me an idea about the overall software coding task.</td>
<td>4.27</td>
<td>1.01</td>
<td>3.63</td>
<td>1.06</td>
</tr>
<tr>
<td>This design provides sufficient detail for me to write the software code for the VR interface.</td>
<td>3.64</td>
<td>1.29</td>
<td>3.13</td>
<td>1.25</td>
</tr>
<tr>
<td>This design provides a nice balance between high-level and low-level design specifications.</td>
<td>4.18</td>
<td>0.75</td>
<td>3.38</td>
<td>1.19</td>
</tr>
<tr>
<td>The components specified in this design are reusable in developing similar VR interface.</td>
<td>3.93</td>
<td>0.61</td>
<td>3.21</td>
<td>0.96</td>
</tr>
<tr>
<td>This design does a good job in specifying the graphical representations of animals in the VR interface.</td>
<td>4.18</td>
<td>0.98</td>
<td>1.38</td>
<td>0.74</td>
</tr>
<tr>
<td>This design does a good job in specifying the behaviors of the animals in the VR interface.</td>
<td>4.18</td>
<td>1.25</td>
<td>3.88</td>
<td>0.99</td>
</tr>
<tr>
<td>This design does a good job in specifying the communications among the animals in the VR interface.</td>
<td>4.09</td>
<td>1.04</td>
<td>3.25</td>
<td>1.49</td>
</tr>
<tr>
<td>This design does a good job in specifying user interactions with animals in the VR interface.</td>
<td>4.09</td>
<td>0.83</td>
<td>3.13</td>
<td>1.36</td>
</tr>
<tr>
<td>This design does a good job in linking animal behaviors with graphical representations that are required for displaying those behaviors.</td>
<td>4.00</td>
<td>0.89</td>
<td>2.25</td>
<td>0.71</td>
</tr>
<tr>
<td>This design would reduce my mental workload in writing the software code for the VR interface.</td>
<td>3.73</td>
<td>0.79</td>
<td>3.5</td>
<td>1.31</td>
</tr>
<tr>
<td>Overall, I find this design successful.</td>
<td>3.91</td>
<td>1.14</td>
<td>3.5</td>
<td>1.07</td>
</tr>
</tbody>
</table>

Table 6.3 Software developer's satisfaction with designs built with VRID and OO models and methodologies

Note: All questions were posed using a 5-point Likert scale ranging from disagree (1) to agree (5). M and SD represent mean and standard deviation respectively.
Modifiability of the resulting designs. We also assessed the degree to which the resulting designs are easy to modify by asking software developers to estimate the time it would take them to make a specified change in the design. The responses indicate that it is easier to modify the designs built with the VRID model and methodology. The mean time to make the required change was 1.27 hours (standard deviation: 0.51) for the designs built with VRID model and methodology whereas it was 1.88 hours (standard deviation: 1.48) for the designs built with the OO model and methodology. We also asked the software developers to estimate the ease of making the required change. As Table 6.4 indicates, software developers found the designs built with the VRID design model and methodology easier to modify. These finding indicate that the use of multi-component object architecture of the VRID model results in modular and easier to modify VR interface designs. In the absence of such a conceptual tool, designers find it more difficult to decompose the design into conceptually distinct parts such as graphics, behaviors, interactions, and communications. When these distinctions are not built into the design, it becomes more difficult to change in the design in the implementation and maintenance phases.

<table>
<thead>
<tr>
<th>Questions</th>
<th>VRID</th>
<th>OO</th>
</tr>
</thead>
<tbody>
<tr>
<td>It would take me a lot of time to modify this design to incorporate the required change in the interface description.</td>
<td>2.18 0.60</td>
<td>2.38 0.92</td>
</tr>
<tr>
<td>It would take me a lot of mental effort to modify this design to incorporate the required change in the interface description.</td>
<td>2.27 0.90</td>
<td>2.75 1.49</td>
</tr>
<tr>
<td>It would be difficult for me to modify this design to incorporate the required change in the interface description.</td>
<td>1.82 0.40</td>
<td>2.25 1.04</td>
</tr>
<tr>
<td>This design makes it difficult to incorporate these kinds of changes in the interface description.</td>
<td>1.73 0.65</td>
<td>2.25 1.04</td>
</tr>
</tbody>
</table>

Table 6.4 Ease of modifiability of designs built with VRID and OO models and methodologies

Notes: All questions were posed using a 5-point Likert scale ranging from disagree (1) to agree (5). M and SD represent mean and standard deviation respectively.
SUMMARY AND DISCUSSIONS

This study allowed us to investigate both designers' and software developers' perspectives about the validity, usability, and usefulness of using the VRID model and methodology. We compared the VRID model and methodology with the well-established object oriented design model and methodology. Findings indicate that the designers using the VRID performed better in designing a VR interface, as measures by the mean time to complete the design. VRID was easier to learn. It provided better guidance in decomposing the overall design task into smaller, less complex, and conceptually distinct parts. It provided better guidance to designers in making design choices and tradeoffs. Overall, designers were more satisfied with the VRID than the OO model and methodology.

Software developers, who were responsible for coding the VR interface based on the designs produced by VRID and OO models and methodologies, were also more satisfied with the designs built with VRID. VRID designs were easier to understand. They provided a higher-level view of the overall coding task for the developers. They achieved a nice balance between the high-level and low-level design specifications, and provided sufficient detail for doing the actual coding. Software developers found the components of the VRID designs more reusable compared to the OO designs. They also thought that the graphical, behavioral, interaction and communication aspects of the interface elements were better specified in the VRID designs. These features also made the VRID designs easier to modify. For the given modification task, software developers estimated
that the VRID designs would take less time and be easier to modify compared to the OO designs. Overall, software developers found the designs built with VRID model and methodology more successful than the designs built with the OO model and methodology.

This study has a few limitations. First, our design task is a simplified VR interface. It may not be representative of complex interfaces of VR applications that will have any practical significance in the real life. Yet, even this simple VR interface took 8.19 hours and 5.82 hours, on average, to design with the OO and VRID models and methodologies respectively. Therefore, it was not feasible in this study to design more complex VR interfaces. Despite its limited representativeness, we believe that our simple VR interface task was appropriate for this study because it contained almost all of the properties that a complex VR interface might contain.

A second limitation of this study is its limited sample size. Our findings above are based on the comparison of descriptive statistics. Due to low sample size, we were not able to run statistical tests to compare the VRID and OO groups. However, it was not feasible to increase the sample size. The two phases of our study required a total commitment of about 10-14 hours per participant. Given the simplicity of our VR interface design task, and the significant time and effort required for design and evaluation even in this simple task, it was a very significant challenge to attract larger number of participants. Increasing the sample size would require offering satisfactory incentives to participants, rather than the limited course credits we were able to offer. Offering satisfactory
incentives depended on availability of significant financial resources, which were beyond the budget of this study. Nevertheless, we believe that the 19 designers we have used in this study constitute a good enough sample size, given that this was the early, exploratory assessment phase of a new model and methodology development study. Clearly, there is need for replicating this user study in the context of more complex VR interface design tasks and with larger sample sizes.
CHAPTER 7: CONCLUSIONS AND FUTURE WORK

In this thesis, we identified a gap in the VR literature, namely, the lack of high-level design models and methodologies for design and development of VR interfaces. By proposing the VRID model and methodology as one possible approach, we have taken an initial step towards addressing this gap. The VRID model provides conceptual guidance to designers in specifying graphical features, behaviors, interactions, and communication mechanisms of interface objects through the use of its multi-component object architecture. The VRID methodology provides procedural steps and guidelines as to how the VRID model can be applied methodically during the design process. Collectively, the VRID model and methodology brings conceptual and methodological rigor to the design of VR interfaces.

We evaluated the VRID model and methodology by applying them to the design of various types and complexities of VR interfaces, which we identified from the literature or created for test purposes. These theoretical evaluations provided preliminary evidence for the validity, usability, and usefulness of the VRID model and methodology. They also helped us to identify and address the shortcomings of the model and methodology. Then, we conducted a user study to investigate the benefits of the VRID model and methodology for designers and software developers. The user study allowed us to benchmark validity, usability, and usefulness of the VRID model and methodology against those of the well-known object oriented (OO) design model and methodology. The findings indicate that the VRID model and methodology are either comparable to or
superior than the OO model and methodology in guiding the design of VR interfaces. These findings provide some empirical evidence for the validity, usability and usefulness of the VRID model and methodology.

The VRID model allows designers to think comprehensively about various types of human-computer interactions, objects, behaviors, and communications that need to be supported by VR interfaces. It enables designers to decompose the overall design task into smaller, conceptually distinct, and easier to design tasks. It also provides a common framework and vocabulary, which can enhance communication and collaboration among users, designers and software developers involved in the development of VR interfaces. The VRID model may also be useful in implementation and maintenance stages of the life cycle of a VR interface since it isolates details of various components of a VR interface through its multi-component object architecture. It makes changes in one component transparent to the other components. Flexibility of the model in terms of types of objects, behaviors, interfaces, and communications supported makes it applicable in a broad range of VR interfaces.

The VRID methodology contributes to practice by guiding designers in the application of the VRID model to the VR interface design process. The methodology formalizes the process of VR interface design into two phases, which represent different levels of abstraction, and breaks down the phases into a discrete number of steps. High-level design phase helps designers to clarify the requirements of the interface. Furthermore, this phase encourages collaboration of designers and users with its simple, easy to
understand steps. The second phase of the methodology, low-level design phase, guides designers in specifying procedural details of the interface. The steps of the low-level design phase offer designers flexibility in choosing languages, methods or mechanisms required to specify fine-grained procedural details of the interface. This makes it easier to apply the methodology to a wide range of VR interfaces.

This thesis constitutes a first step towards providing high-level design models and methodologies for the design and development of VR interfaces. Creation of new knowledge, models, and methodologies in this area is crucial for further advancement of the VR field, and for the proliferation and diffusion of VR applications in practice. Further work is required to test the robustness and generalizability of the VRID model and methodology across various types of VR projects in the industry. Our user study needs to be replicated in the context of more complex VR interfaces and with larger sample sizes. Once the validity, usability, and usefulness of VRID model and methodology are established across a variety of complex VR interfaces, there will be a need to develop a user interface design tool that can facilitate the use of the VRID methodology.
APPENDIX A: EVALUATIONS OF DESIGN MODEL AND METHODOLOGIES

1. THE VRID MODEL AND METHODOLOGY

Questionnaire 1

In this questionnaire, we would like to learn about your current level of knowledge about the VRID Model and the VRID methodology. Please indicate your level of agreement or disagreement with the following statements by circling the number of your choice.

<table>
<thead>
<tr>
<th>Statement</th>
<th>Slightly Disagree</th>
<th>Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. In general, I understand the distinction between a design model and a design methodology.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>2. I can see the distinction between the VRID Model and the VRID Methodology.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>VRID MODEL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. I understand the purpose of the graphics layer in the VRID Model.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>4. I understand the purpose of the behavioral layers in the VRID Model.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>5. I understand the purpose of the interaction layer in the VRID Model.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6. I understand the purpose of the mediator layer in the VRID Model.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>7. I understand the purpose of the communication layer in the VRID Model.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>8. The distinctions among physical, magical, and composite behaviors are clear to me.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>9. The distinction between internal and external communications of an object is clear to me.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>10. Overall, I understand the 7-layer architecture of the VRID Model.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>11. Overall, I can see the benefits of using the VRID Model in developing VR interfaces.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>12. Currently, how knowledgeable do you feel about using the VRID Model in designing</td>
<td>Not very knowledgeable</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Moderately Knowledgeable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Very Knowledgeable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

91
a VR interface?

### VRID METHODOLOGY

1. I understand why the VRID methodology breaks up the design process into **high-level** and **low-level** design phases.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

2. I understand the **high-level design activities** outlined in the VRID methodology.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

3. I understand the **low-level design activities** outlined in the VRID methodology.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

4. I learned **how to identify data elements** in a VR interface.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

5. I learned **how to identify objects** in a VR interface.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

6. I learned **how to model/specify graphics** of objects.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

7. I learned **how to model/specify behaviors** of objects.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

8. I learned **how to model/specify interactions** of objects.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

9. I learned **how to model/specify internal communications** of objects.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

10. I learned **how to model/specify external communications** of objects.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

11. Overall, I know **how to apply the VRID Model** in designing a virtual reality interface.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

12. Overall, I know **how to use the VRID Methodology** in designing a virtual reality interface.

<table>
<thead>
<tr>
<th></th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

13. Overall, I can see the **benefits of using the VRID Methodology** in designing virtual reality interfaces.

<table>
<thead>
<tr>
<th></th>
<th>Not very knowledgeable</th>
<th>Moderately Knowledgeable</th>
<th>Very Knowledgeable</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

14. Currently, how knowledgeable do you feel about using the **VRID Methodology** in designing a VR interface?

<table>
<thead>
<tr>
<th></th>
<th>Not very knowledgeable</th>
<th>Moderately Knowledgeable</th>
<th>Very Knowledgeable</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
**Questionnaire 2**

The purpose of this questionnaire is to evaluate appropriateness and usefulness of the VRID design methodology in designing Virtual Reality (VR) interfaces. Please fill in this questionnaire **ONLY AFTER** completing the design of the assigned VR interface.

1. In total, how many hours did it take you to complete the assigned VR interface design?
   ____ hours

2. To what extent was the description of the VR interface CLEAR?

<table>
<thead>
<tr>
<th>Not clear at all</th>
<th>Slightly Clear</th>
<th>Moderately Clear</th>
<th>Clear</th>
<th>Very Clear</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

3. To what extent was the description of the VR interface SUFFICIENT for doing the design?

<table>
<thead>
<tr>
<th>Not at all Sufficient</th>
<th>Slightly Sufficient</th>
<th>Moderately Sufficient</th>
<th>Sufficient</th>
<th>Very Sufficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

Please indicate your level of agreement or disagreement with the following statements based on your experience in designing the assigned VR interface. Please circle the number of your choice.

4. **Learning to use** the VRID design methodology was easy for me.

<table>
<thead>
<tr>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

5. **Using** the VRID design methodology required a lot of mental effort.

   | 1 | 2 | 3 | 4 | 5 |

6. The VRID design methodology provided guidance in **decomposing the overall design task** into smaller, less complex, and conceptually distinct parts.

   | 1 | 2 | 3 | 4 | 5 |

7. The VRID design methodology enabled me to **understand various design choices and tradeoffs**.

   | 1 | 2 | 3 | 4 | 5 |

8. It was **easy for me to use** the VRID design methodology in designing the assigned VR interface.

   | 1 | 2 | 3 | 4 | 5 |

9. The VRID design methodology **met my needs** in designing the assigned VR interface.

   | 1 | 2 | 3 | 4 | 5 |
Open-ended questions
1. Please explain any major difficulties that you might have faced in designing this Virtual Reality interface.

2. Based on your experience in this assignment, how would you describe the advantages of the VRID methodology in designing Virtual Reality interfaces?

3. Based on your experience in this assignment, how would you describe the disadvantages and shortcomings of the VRID methodology in designing Virtual Reality interfaces?
Demographics

1. Please enter your age: _______ years

2. Your gender (please check): ___Male  ___Female

3. Please check your year in Tufts:
   ___First Year    ___Sophomore    ___Junior    ___Senior
   ___Graduate

4. Please check your Major:
   ___Elec. Eng.    ___Comp. Sci.    ___Comp. Eng.    ___Other (Please specify): _____

5. Please list the computer languages that you know well.

A software design project refers to a project, which involves high-level partitioning of the design problem, drawing of the design diagrams, and specification of software components and how they will work together.

6. How many software design projects have you done in your coursework?
   ____ projects in total.

Please list or describe all major software design projects that you have done in your coursework:
7. How many software design projects have you done in the industry?
   ____ projects in total.

Please list or describe all major software design projects that you have done in industry:

8. Do you have any design experience with virtual reality software? (please check)
   ____No  ____Yes (Please explain briefly:)

96
The Results of Questionnaire 1

**Questions** | M  | SD  
--- | --- | ---  
In general, I understand the distinction between a design model and a design methodology. | 3.91 | 1.14  
I can see the distinction between the VRID Model and the VRID Methodology. | 3.82 | 1.08  
I understand the purpose of the graphics layer in the VRID Model. | 4.55 | 0.69  
I understand the purpose of the behavioral layers in the VRID Model. | 4.45 | 0.69  
I understand the purpose of the interaction layer in the VRID Model. | 4.27 | 0.79  
I understand the purpose of the mediator layer in the VRID Model. | 4.27 | 0.65  
I understand the purpose of the communication layer in the VRID Model. | 4.18 | 0.60  
The distinctions among physical, magical, and composite behaviors are clear to me. | 4.36 | 0.67  
The distinction between internal and external communications of an object is clear to me. | 3.73 | 1.00  
Overall, I understand the 7-layer architecture of the VRID Model. | 3.91 | 0.70  
Overall, I can see the benefits of using the VRID Model in developing VR interfaces. | 3.55 | 1.44  
Currently, how knowledgeable do you feel about using the VRID Model in designing a VR interface? | 2.91 | 0.94  
I understand why the VRID methodology breaks up the design process into high-level and low-level design phases. | 4.27 | 1.00  
I understand the high-level design activities outlined in the VRID methodology. | 4.18 | 0.87  
I understand the low-level design activities outlined in the VRID methodology. | 3.73 | 0.65  
I learned how to identify data elements in a VR interface. | 4.18 | 0.75  
I learned how to identify objects in a VR interface. | 4.18 | 0.60  
I learned how to model/specify graphics of objects. | 4.18 | 0.75  
I learned how to model/specify behaviors of objects. | 4.09 | 0.7  
I learned how to model/specify interactions of objects. | 3.91 | 0.7  
I learned how to model/specify internal communications of objects. | 3.55 | 0.82  
I learned how to model/specify external communications of objects. | 3.64 | 0.81  
Overall, I know how to apply the VRID Model in designing a virtual reality interface. | 3.55 | 0.93  
Overall, I know how to use the VRID Methodology in designing a virtual reality interface. | 3.73 | 0.90  
Overall, I can see the benefits of using the VRID Methodology in designing virtual reality interfaces. | 3.55 | 1.44  
Currently, how knowledgeable do you feel about using the VRID Methodology in designing a VR interface? | 2.73 | 1.00  

97
Notes: All questions were posed using a 5-point Likert scale ranging from disagree (1) to agree (5). M and SD represent mean and standard deviation respectively.

2. OBJECT ORIENTED DESIGN MODEL AND METHODOLOGY

Questionnaire 1

In this questionnaire, we would like to learn about any prior experience that you might have had in using the object-oriented design methodology. Please fill out this questionnaire AFTER reading the assigned VR interface description, but BEFORE starting with the analysis and design of the interface.

1. When did you get your first formal training in the object-oriented design methodology?
   ____ In this class (Comp180).
   ____ Before this class (Comp180). Please specify when: ___(month)/___(year)

2. How many class projects have you done using the object-oriented design methodology?
   ____ projects.

3. How many outside of class projects have you done using the object oriented design methodology?
   ____ projects.

4. Overall, how long have you been using the object-oriented design methodology?
   ____ days / months / years (please circle the appropriate unit of time)

5. Currently, how knowledgeable do you feel about using the object-oriented design methodology to design a VR interface?

<table>
<thead>
<tr>
<th>Not very knowledgeable</th>
<th>Moderately Knowledgeable</th>
<th>Very Knowledgeable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

98
Questionnaire 2

The purpose of this questionnaire is to evaluate appropriateness and usefulness of object-oriented design methodology in designing Virtual Reality (VR) interfaces. Please fill in this questionnaire ONLY AFTER completing the design of the assigned VR interface.

1. In total, how many hours did it take you to complete the assigned VR interface design? 
   ____ hours

2. To what extent was the description of the VR interface CLEAR? 
   Not clear at all 1  Slightly Clear 2  Moderately Clear 3  Clear 4  Very Clear 5

3. To what extent was the description of the VR interface SUFICIENT for doing the design? 
   Not at all Sufficient 1  Slightly Sufficient 2  Moderately Sufficient 3  Sufficient 4  Very Sufficient 5

Please indicate your level of agreement or disagreement with the following statements based on your experience in designing the assigned VR interface. Please circle the number of your choice.

4. Learning to use the object oriented design methodology was easy for me. 
   Disagree 1  Slightly Disagree 2  Neutral 3  Slightly Agree 4  Agree 5

5. Using the object oriented design methodology required a lot of mental effort. 
   1  2  3  4  5

6. The object oriented design methodology provided guidance in decomposing the overall design task into smaller, less complex, and conceptually distinct parts. 
   1  2  3  4  5

7. The object oriented design methodology enabled me to understand various design choices and tradeoffs. 
   1  2  3  4  5

8. It was easy for me to use the object oriented design methodology in designing the assigned VR interface. 
   1  2  3  4  5

9. The object oriented design methodology met my needs in designing the assigned VR interface. 
   1  2  3  4  5

Open-ended questions
   1. Please explain any major difficulties that you might have faced in designing this Virtual Reality interface.

99
2. Based on your experience in this assignment, how would you describe the advantages of object-oriented methodology in designing Virtual Reality interfaces?

3. Based on your experience in this assignment, how would you describe the disadvantages and shortcomings of object-oriented methodology in designing Virtual Reality interfaces?
Demographics

1. Please enter your age: _______ years

2. Your gender (please check): ___ Male ___ Female

3. Please check your year in Tufts:
   ___ First Year ___ Sophomore ___ Junior ___ Senior
   ___ Graduate

4. Please check your Major:
   ___ Elec. Eng. ___ Comp. Sci. ___ Comp. Eng. ___ Other (Please specify): ___

5. Please list the computer languages that you know well.

A software design project refers to a project, which involves high-level partitioning of the design problem, drawing of the design diagrams, and specification of software components and how they will work together.

6. How many software design projects have you done in your coursework?
   _____ projects in total.

Please list or describe all major software design projects that you have done in your coursework:
7. How many **software design projects** have you done in the industry?

___ projects in total.

Please list or describe all major software design projects that you have done in industry:


8. Do you have any design experience with virtual reality software? (please check)

___ No

___ Yes (Please explain briefly:)


102
APPENDIX B : EVALUATIONS OF DESIGN SOLUTIONS

1. THE VRID MODEL AND METHODOLOGY

Evaluation Questionnaire-1

Please review the design given at the end of this packet before filling out this survey. Suppose that you are asked to write the software code for the VR interface based on the specifications given in this design.

Please indicate your level of agreement or disagreement with the following statements by circling the number of your choice.

<table>
<thead>
<tr>
<th>Statement</th>
<th>Disagree</th>
<th>Slightly Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. This design is easy to understand.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>2. This design provides a high-level view of the VR interface and gives me an idea about the overall software coding task.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>3. This design provides sufficient detail for me to write the software code for the VR interface.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>4. This design provides a nice balance between high-level and low-level design specifications.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>5. The specifications that are developed for data elements specified in this design are reusable in similar VR interface designs.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6. The specifications that are developed for object behaviors specified in this design are reusable in similar VR interface designs.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>7. The specifications that are developed for User and Object interactions in this design are reusable in similar VR interface designs.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>
11. The specifications that are developed for the *internal communications of the objects* in this design are *reusable* in similar VR interface designs.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disagree</td>
<td>Slightly Disagree</td>
<td>Neutral</td>
<td>Slightly Agree</td>
<td>Agree</td>
</tr>
</tbody>
</table>

12. The specifications that are developed for the *external communications of the objects* in this design are *reusable* in similar VR interface designs.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disagree</td>
<td>Slightly Disagree</td>
<td>Neutral</td>
<td>Slightly Agree</td>
<td>Agree</td>
</tr>
</tbody>
</table>

13. This design does a good job in specifying the *graphical representations* of animals in the VR interface.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disagree</td>
<td>Slightly Disagree</td>
<td>Neutral</td>
<td>Slightly Agree</td>
<td>Agree</td>
</tr>
</tbody>
</table>

14. This design does a good job in specifying the *behaviors* of the animals in the VR interface.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disagree</td>
<td>Slightly Disagree</td>
<td>Neutral</td>
<td>Slightly Agree</td>
<td>Agree</td>
</tr>
</tbody>
</table>

15. This design does a good job in specifying the *communications* among the animals in the VR interface.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disagree</td>
<td>Slightly Disagree</td>
<td>Neutral</td>
<td>Slightly Agree</td>
<td>Agree</td>
</tr>
</tbody>
</table>

16. This design does a good job in specifying *interactions of the user* with the animals in the VR interface.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disagree</td>
<td>Slightly Disagree</td>
<td>Neutral</td>
<td>Slightly Agree</td>
<td>Agree</td>
</tr>
</tbody>
</table>

17. This design does a good job in linking animal *behaviors with graphical representations* that are required for displaying those behaviors.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disagree</td>
<td>Slightly Disagree</td>
<td>Neutral</td>
<td>Slightly Agree</td>
<td>Agree</td>
</tr>
</tbody>
</table>

18. This design would reduce my mental workload in *writing the software code* for the VR interface.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disagree</td>
<td>Slightly Disagree</td>
<td>Neutral</td>
<td>Slightly Agree</td>
<td>Agree</td>
</tr>
</tbody>
</table>

19. Overall, I find this design *successful*.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disagree</td>
<td>Slightly Disagree</td>
<td>Neutral</td>
<td>Slightly Agree</td>
<td>Agree</td>
</tr>
</tbody>
</table>
20. Are there any errors in this design? Please specify.

21. Are there any missing features in this design? Please specify.
Changes in VR Interface Description

Suppose that the project owner wants to make the following changes in the last paragraph of the VR interface description:

Original paragraph:

[...]
If the user moves his/her hand towards the spider, initially the spider stands still. If the distance between the hand and the spider becomes shorter than a predefined distance, the spider starts moving in the opposite direction of the user’s hand movement: i.e., if the hand moves to the right, the spider moves to the left. If the hand moves to the left, the spider moves to the right. If the spider comes closer to the parrot than a predefined distance, the parrot stops drinking water from the cup. If the user stops moving his/her hand, the spider stops moving.

New paragraph:

[...]
If the user moves his/her hand towards the spider, initially the spider stands still. If the distance between the hand and the spider becomes shorter than a predefined distance, the spider starts jumping up and down in the opposite direction of the user’s hand movement: i.e., if the hand moves to the right, the spider jumps up and down to the left. If the hand moves to the left, the spider jumps up and down to the right. If the spider comes closer to the parrot than a predefined distance, the parrot stops drinking water from the cup and looks up and down to observe the spider’s movements.
Please briefly describe how you would modify this particular design to incorporate the required changes in the interface description:


Please check the sentence that best describes the level of effort required for incorporating the required change into this particular design:

☐ I would have to make very minor changes in this design to incorporate the required change.
☐ I would have to make minor changes in this design to incorporate the required change.
☐ I would have to make moderate changes in this design to incorporate the required change.
☐ I would have to make substantial changes in this design to incorporate the required change.
☐ I would have to change this design completely to incorporate the required change.

Please estimate how many hours it would take you to incorporate the required change into this particular design?

__________ hours.
Please indicate your level of agreement or disagreement with the following statements by circling the number of your choice.

1. It would **take me a lot of time to modify this design** to incorporate the required change in the interface description.

2. It would **take me a lot of mental effort to modify this design** to incorporate the required change in the interface description.

3. It would be **difficult for me to modify this design** to incorporate the required change in the interface description.

4. This design makes it **difficult to incorporate these kinds of changes in the interface description**.
2. OBJECT ORIENTED DESIGN MODEL AND METHODOLOGY

Evaluation Questionnaire-1

Please review the design given at the end of this packet before filling out this survey. Suppose that you are asked to write the software code for the VR interface based on the specifications given in this design.

Please indicate your level of agreement or disagreement with the following statements by circling the number of your choice.

<table>
<thead>
<tr>
<th></th>
<th>Slightly Disagree</th>
<th>Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. This design is easy to <strong>understand</strong>.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>2. This design <strong>provides a high-level view of the VR interface</strong> and gives me an idea about the overall software coding task.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>3. This design provides <strong>sufficient detail</strong> for me to write the software code for the VR interface.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>4. This design provides a nice <strong>balance between high-level and low-level design specifications</strong>.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>5. <strong>The object attributes</strong> specified in this design are <strong>reusable</strong> in developing similar VR interfaces.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6. <strong>The object methods</strong> specified in this design are <strong>reusable</strong> in developing similar VR interfaces.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>7. <strong>The messages</strong> specified in this design are <strong>reusable</strong> in developing similar VR interfaces.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>8. This design does a good job in specifying the <strong>graphical representations</strong> of animals in the VR interface.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>9. This design does a good job in specifying the <strong>behaviors</strong> of the animals in the VR interface.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>
10. This design does a good job in specifying the **communications** among the animals in the VR interface.

11. This design does a good job in specifying **user interactions** with animals in the VR interface.

12. This design does a good job in linking **animal behaviors with graphical representations** that are required for displaying those behaviors.

13. This design would reduce my mental workload in **writing the software code** for the VR interface.

14. Overall, I find this design **successful**.

15. Are there any errors in this design? Please specify.

16. Are there any missing features in this design? Please specify.
Evaluation Questionnaire-2

Changes in VR Interface Description

Suppose that the project owner wants to make the following changes in the last paragraph of the VR interface description:

Original paragraph:

[...]
If the user moves his/her hand towards the spider, initially the spider stands still. If the distance between the hand and the spider becomes shorter than a predefined distance, the spider starts moving in the opposite direction of the user’s hand movement: i.e., if the hand moves to the right, the spider moves to the left. If the hand moves to the left, the spider moves to the right. If the spider comes closer to the parrot than a predefined distance, the parrot stops drinking water from the cup. If the user stops moving his/her hand, the spider stops moving.

New paragraph:

[...]
If the user moves his/her hand towards the spider, initially the spider stands still. If the distance between the hand and the spider becomes shorter than a predefined distance, the spider starts jumping up and down in the opposite direction of the user’s hand movement: i.e., if the hand moves to the right, the spider jumps up and down to the left. If the hand moves to the left, the spider jumps up and down to the right. If the spider comes closer to the parrot than a predefined distance, the parrot stops drinking water from the cup and looks up and down to observe the spider’s movements.
Please briefly describe how you would modify this particular design to incorporate the required changes in the interface description:

Please check the sentence that best describes the level of effort required for incorporating the required change into this particular design:

- I would have to make **very minor changes** in this design to incorporate the required change.
- I would have to make **minor changes** in this design to incorporate the required change.
- I would have to make **moderate changes** in this design to incorporate the required change.
- I would have to make **substantial changes** in this design to incorporate the required change.
- I would have to **change this design completely** to incorporate the required change.

Please estimate how many hours it would take you to incorporate the required change into this particular design?

_______ hours.
Please indicate your level of agreement or disagreement with the following statements by circling the number of your choice.

<table>
<thead>
<tr>
<th></th>
<th>Slightly Disagree</th>
<th>Disagree</th>
<th>Neutral</th>
<th>Slightly Agree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. It would <strong>take me a lot of time to modify this design</strong> to incorporate the required change in the interface description.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>2. It would <strong>take me a lot of mental effort to modify this design</strong> to incorporate the required change in the interface description.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>3. It would be <strong>difficult for me to modify this design</strong> to incorporate the required change in the interface description.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>4. This design makes it <strong>difficult to incorporate</strong> these kinds of changes in the interface description.</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>
REFERENCES


