Framework for Context-Aware Information Processing for Design Review in a Virtual Environment

Mohd Fairuz Shiratuddin

Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
In
Environmental Design and Planning

Dr. Walid Thabet, Chair
Dr. Yvan Beliveau
Dr. Kihong Ku
Dr. John Messner

(Feb 6th, 2009)
Blacksburg, Virginia

Keywords: 3d, design review, game engine, virtual environment

Framework for Context-Aware Information Processing for Design Review in a Virtual Environment

Abstract

Design review is a process of reviewing construction design documents to ensure that they reflect the owner's design intent, and are accurate in describing the owner's desired building or facility. Information generation becomes more intensive as the design stage progresses. The use of valuable information during design review stage can lead to a more comprehensive and high quality design, and a building or facility that is constructible, and within the intended budget. However, in current design practices, valuable design review information is scattered, ineffectively placed, and is not used efficiently. The design review process will be more efficient if this valuable information is integrated and centralized.

The author developed a framework to improve the design review process by incorporating a centralized repository of design review information and 3D CAD model, in an interactive Virtual Environment (VE). To develop the framework, the author used Action Research style where he identified and confirmed the design review problem area, promoted the potential solutions to the problem, and developed a prototype. In gathering and analyzing the data for the research, the author used the synthesis of three methods. They include review of literature, a case study (interviews with industry personnel and content analysis of design review documents), and dissemination of the author's progressive findings in conferences, conference proceedings and journal publications.

From his findings, the author developed the framework to improve the design review process by using information filtering based on context-aware concept, coupled with the benefits of a VE. The required design review information in the form of textual, numerical and geometric information is processed (queried, retrieved and stored). The author defined four contexts for information filtering: discipline-centric, task-centric, object-centric, and location-centric. IF-THEN rules are used to trigger the processing of the required design review information and present it to the design reviewer in a VE. A low cost 3D Game Engine is used as the enabling development tool to develop a work-in-progress (WIP) prototype design review application in a VE.

ACKNOWLEDGMENT

I would like to sincerely express my utmost gratitude my adviser, Dr. Walid Thabet for being a great mentor and provide his endless assistance, encouragement and patience in guiding me to complete this dissertation.

To Professor Yvan Beliveau who has been providing his experience and wisdom, without which, this dissertation may not be where it is now.

To Dr. John Messner and Dr. Kihong Ku for providing invaluable guidance with fresh and different perspectives.

To my loving family who has been there for me since day one. To my dearest wife, Sila and my beautiful and inspirational children; Hikmah, Haneef, Hidayah, Haniyya and Hadi.

To all my friends who were students at Virginia Tech, whom together we stood the face of challenges, and provide support to one another; especially to Jose Perdomo, Brendan Johnston and Svetlana Olbina.

…live long & prosper…

TABLE OF CONTENT

APPENDICES 151

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

1 INTRODUCTION TO THE RESEARCH

1.1 Introduction

In a typical construction project, large quantities of valuable information are produced during the design stage. Information generation becomes more intensive as the design stage progresses. Studies by Lutz (1990), East *et al.* (1995a), East *et al.* (1995b), East (1998), Spillinger (2000), and East *et al.* (2004) show the use of valuable information during this stage can influence the quality and total cost of a facility. This can be achieved not only through the design process but also the design review process. An effective design review process allows for a more comprehensive; constructible and high quality design; and construction documents that in turn result in construction cost that is within the intended budget. Spillinger (2000) in a report for the Federal Facility Council (FFC) stated the benefits that can be achieved by effective design review. The benefits include:

a) less rework by the construction contractor
b) fewer change orders to the owner for correction of design errors or omissions, and the cost of belatedly adding project upgrade features that should have been addressed in the original design
c) avoiding costs associated with loss of productivity during construction-delayed facility start-up, and litigation

Spillinger concluded that effective design review increases the likelihood that a business need will be successfully supported by a facility that was thought, designed, constructed, and placed into operation efficiently and effectively.

A design review process is used by the Architect, Engineer and Contractor (AEC) community to help ensure the quality of designs (East, 1998). The design review process reviews the materials to use; the building systems components (structural, mechanical, electrical, plumbing etc.); compliance with the current codes and standards; and the maintenance requirements. Design review process is important for detecting and identifying differences, errors and inconsistencies in designs (East *et al.* 1995a; East *et al.* 1995b, East 1998; Spillinger 2000; East *et al.* 2004). Such deficiencies are expected because the designs are prepared by various design professionals.

Defective designs bring adverse impact on project performances and participants (Andi & Minato, 2003), and are responsible for many of construction failures and result to change orders during construction stage. Change orders have long been identified to have a negative impact on construction productivity, affecting labor efficiency and, occasionally, sizable loss of man hours (Barrie & Paulson 1996; Moselhi 1998). Andi and Minato (2003) reported an example of a simple but costly mistake by the Nikkei Construction in Japan in 2001. A designer unintentionally looked at a wrong number in a standard table and adopted it to design a conduit pipe as one element of a pumping facility. Because of this small and insignificant error, the client had to spend more than 5 million yen (roughly 45,259.11 USD at the currency exchange rate of 1 USD = 110.475 JPY of August 2008) for construction rework.

Lutz (1990) defined “design deficiencies” as conflicts, omissions, or errors in the design documents that go undetected during the design phase. Andi and Minato (2003) defined defective design as product design that does not conform to the acceptable level of quality as required by the owners, contractors or applicable codes and regulations. Defects found in design documents may range from simple miscalculations or omissions in the drawings to systematic errors that lead to critical failures. Many faults in buildings can be traced back to incomplete and inaccurate information, and the inability to use information that has been provided. Conflicts between drawings and specifications can and do lead to conflict. Often relevant information may not be available when needed. This lack of information leads to a constant reinventing of the wheel (Veshosky, 1998). Even if information is available and shared, the meaning of the message may not be correctly interpreted or understood (Pietroforte, 1997). Emmitt and Gorce (2003) believed that some of the problems can be avoided with well-designed management systems providing useful information and information coordination.

According to East (1998) design review is still imperfect despite the current tools available because reviewers are subject to several pressures in the design review process. First, design review is inherently a time-consuming process. Second, design review is a resource-constrained process; assigning experts to “more critical” design and construction tasks creating resource bottlenecks. Third, time constraints because of backlogs of unreviewed drawings and specifications may force reviewers to sacrifice the thoroughness of their reviews. Also, time constraints on project funding cycles constrain the ability of the designer to incorporate changes that have been recommended by design reviewers. As a result, jobs may be bid with known mistakes. East believed the potential of design review is promising but current methods are still imperfect thus causing the benefits to be unattainable.

This research theorizes that design review process can be improved if valuable and presently scattered information is integrated and centralized. Information filtering based on context-aware concept coupled with the benefits of a VE can unify the presently “dispersed” and “scattered” design review information during design review. The use of context-aware concept allows for access of various design review information categories in real-time. The information is then visually presented in real-time to the respective design reviewer at the right time, in the location for the right object. A low cost 3D Game Engine is used as the enabling development tool to develop a work-in-progress (WIP) prototype design review application in a VE. The WIP prototype design review application is not complete but included workable features that are proposed in this research. Features implemented in the WIP prototype design review application can be found in Appendix 6.

1.2 Problem Statement

From study of the literature and a series of interviews conducted by the author (with local AEC companies, Virginia Tech's Office of the University Architect (OUA), and Virginia Tech's Capital Design and Construction Department (CDCD)), the following research problems were identified.

Problem 1) During design review, much information is referred to by the reviewer. Regulations, codes, checklists, basis of design documents, standards, and manuals are often referred to ensure the design produced follows the project intent and specifications (Figure 1-1**). These large amount of design review related information are scattered (in electronic form or physical papers, documents, books, manuals etc.), making it harder for reviewer to seek this interrelated information quickly and efficiently within a limited review time frame.**

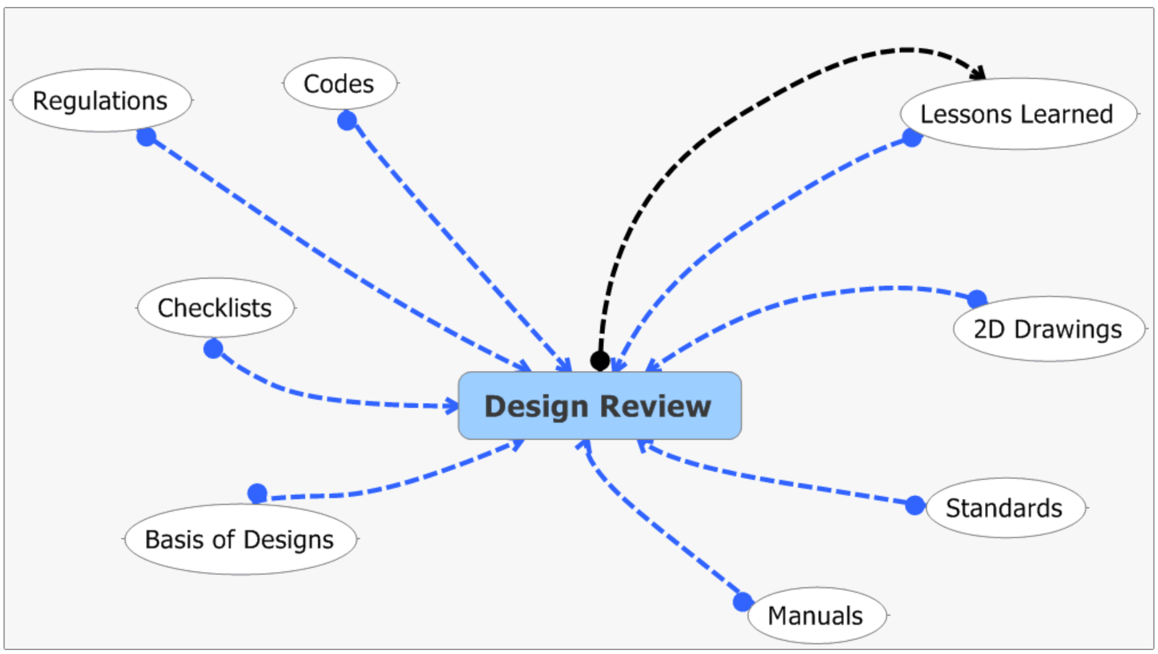


Figure 1-1 - The various types of information required for design review (Shiratuddin & Thabet, 2007)

The complexity of today's construction projects requires the involvement of various specialties (disciplines such as electrical, mechanical, plumbing etc.). To realize the final building or facility requires the production of hundreds of 2D construction drawings, and related project documentation such as contract document, specification etc. The design stage has high chance of information overload because this is when the project participants contributing the key expertise in each stage (Haksever, 2000). Similarly, codes and regulations governing the projects are also referred to. While performing design review, the reviewer has difficulty coordinating many sources of information needed for the design review process.

A case study by Shiratuddin and Thabet (2007) on design review documents has identified the important information requirements for design review. The classifications of information needed to perform design review and shown in Figure 1-1 were arrived at through a series of interviews conducted with AEC companies in Blacksburg and Roanoke, Virginia. The interviews were conducted to identify the general trends of design review, subjects' view on design review in a VE, and the subjects' wish-list/suggestion/recommendations on areas of design review that needs improvement etc. Because of design review information being scattered and the iterative nature of the design review process, many design conflicts and errors go undetected even after a lengthy design review has been performed. From the visits to design practitioners and reviewers' offices, undetected conflicts and errors in design affect the owner or end user. They affect one or more of the following items during construction phase and operational life of the facility - owner satisfaction, schedule and completion, construction and operational costs, user safety and health, user morale, and litigation and liability.

In the construction industry, information is exchanged in the form of either paper-based, or/and electronically. Traditional information-sharing methods (i.e. 2D paper drawings and specifications) have not proven useful in transferring design intent from design to construction (de la Garza & Oralkan, 1995). Although computer-aided design (CAD) is widely used in the industry, the exchange of information is still at large a manual process, particularly for design review. 2D paper drawings are heavily used to identify errors, inconsistencies or conflicts. This process is time-consuming and inefficient. Information is "scattered" as reviewers have to manually search for standards for a particular subsystem and product type, or cross-check 2D construction drawings with other related information (specifications, building codes and regulations, construction manual etc.). The inefficient handling and the "scattered" information can lead to the following design deficiencies (Lutz, 1990):

- Contract document conflicts: conflicts between drawings and specifications
- Interdisciplinary coordination errors: conflict or interference problems between structural, mechanical and electrical
- Technical compliance inconsistencies: non-adherence to the proper design guidelines, technical specifications and building codes

Relying on manual input and output of information during design review is inefficient. This is because 1) errors and inconsistencies may be created or left undetected, 2) record keeping lacks facilitation, and 3) results are not replicated, coordinated and updated automatically (Staub-French *et al.*, 2001).

There is thus a need for design review related information to be centralized, easily accessible and intelligently processed to assist reviewer during review sessions. East *et al.* (2004) suggested that a successful automated design review system will reduce the number of reviewers and the time necessary to complete the review phases, thus, shorten the facility delivery cycle. Reduced delivery duration will provide the owner with favorable earlier occupancy. Earlier occupancy translates into significant total project savings to the owner.

Problem 2) Current design review approaches rely heavily on 2D documentation. 2D documentation is often inadequate a) in representing the owner's intended facility in its entirety, and b) as a design communication tool in conveying the design intent among project team members, and potentially with the owner. Because of this, misinterpretations of design, and errors and inconsistencies in design are unavoidable.

Starting from the owner's requirement and description of the desired facility, 2D drawings are designed and produced by the designer based on the 3D mental model visualized in his or her mind (Dunston *et al.* 2003; de la Garza & Oralkan 1995). It is then the task of the reviewer during design review, and other project team members to interpret the 2D drawings. Even though the 2D drawings are packed with information (in the form of geometric, numerical and textual information), many reviewers will still find it difficult to understand the 2D drawings. The process of interpreting 2D drawings is not perfect as 2D drawings do not adequately represent the multispatial information of the building in a more intuitive way, as 3D can. Only the designer can visualize what the 2D drawings represent and how they look like in 3D.

"Designers spend too much time engaged in the act of drawing that they sometimes forget that reading a drawing and understanding it fully takes much skill and experience" (Emmitt & Gorse, 2003). Other people who lack skill and experience may misread and misinterpret the drawings. Drawings are produced by specialty designers and used by many different disciplines, during both the design process and construction, what is clear to one person may be less so to another. Thus, it is not guaranteed that a drawing will be clear to every viewer. The designer as the originator should be aware of the risk of confusion, misunderstanding, and misinterpretation, thus should ensure that drawings are easy to understand and interpret (Emmitt & Gorse, 2003). Thus teams must spend much time mentally relating project information to support decision-making tasks (Liston *et al.*, 2000).

Various methods are used in performing design review such as interdisciplinary checklists, light tables, online review systems, and physical mock-ups (Shiratuddin & Thabet, 2003a). However, these methods still use 2D designs/drawings as the base for performing design review. In a unique approach of design review, a national modular home builder uses a life-sized physical mock-up (PMU) for their design review. This approach can be helpful for reviewers to physically view and experience, finding errors and conflicts in real space. Reviewers and other project team members can discuss, identify, make comments on the design, and take notes on all the agreed changes that are required before the designs are finalized. Similar to other approaches, 2D drawings are still referred to and compared with the PMU.

Thus, there is a need to represent designs not just in 2D but 3D, where designers, reviewers, and other project team members can view the same model of a facility. Designers can also be more certain the designs they produce is what the owner envisages. Using VE technology can improve the 3D representation of the design that will assist and can provide a common language for project stakeholders.

Problem 3) Current design review approaches do not allow for real-time collaboration, communication and information sharing within the same 3D space.

In the manufacturing industry, sequential manufacturing is the traditional production method used in manufacturing goods (Syan & Menon 1994; Ichida 1996). The manufacturing industry has experience and document many disadvantages of sequential ideas of product manufacturing. Decisions made by designers during the conceptual design stage often have costly effects on later processes and require product redesign or return of the product to previous stages for corrections (Syan & Menon, 1994). This situation is what been experienced by construction industry in the design process. In many situations the early design of a product, whether construction or manufacturing, is often an extensive procedure and has extensive influence in the end product. Blanchard *et al.* (1995) highlighted how the approach of “design it now and fix it later” is costly for companies.

Comparing both sequential processes in manufacturing and construction, sequential processes requires that each stage of production to be taken in turn; the next step in the sequence does not start until completion of the previous step (Ichida, 1996). This process is commonly called the over the wall approach. In sequential manufacturing, design works in relative isolation. The result of the process is that decisions made by designers in the conceptual design stage often produce undesirable effects on later processes. This situation requires product redesign thus causes delays and increases costs (Syan & Menon, 1994).

In general, current design review approaches are iterative and sequential (Figure 1-2), and are executed in 4 consecutive stages following the end of each design phase (Programming, Schematic, Preliminary and Working Drawings). In each design phase, the designer is responsible for producing the drawings, and drawings are handed over to reviewers for comments and recommendations for possible errors and inconsistencies that were found. Once the comments and recommendations are made, the drawings are passed back to the designers for redesign and for incorporating the required changes.

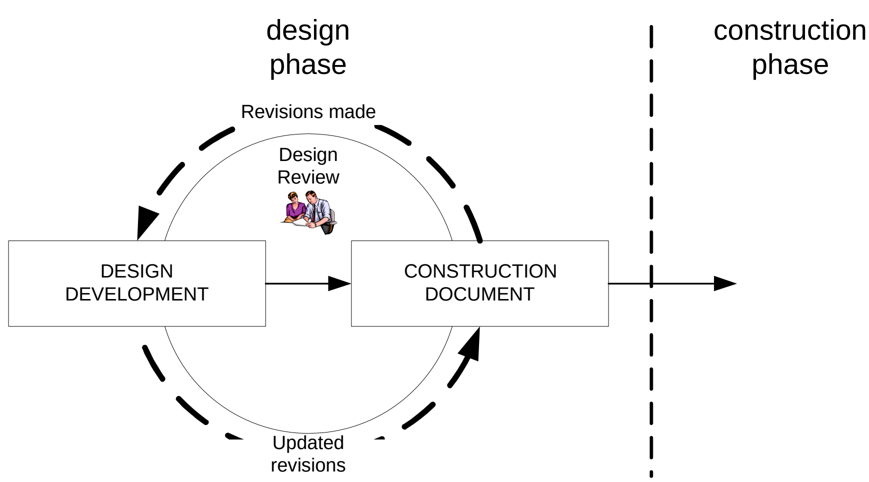


Figure 1-2 - Iterative nature of Design Review

The process of design, review and redesign is cyclical in nature, until satisfactory drawings are accomplished and approved. Once approved the drawings for that design phase are considered complete. The designer can start with the next design phase and finally prepare for construction. A Project Manager or design review coordinator ensures the correct flow of documents and communications among parties involved.

Redesigns will often have knock-on effects and force redesign in others. The time and efforts required for redesigning later of design exceed those involved when the final design is achieved at an early stage. Collaborating and involving all entities; designers, reviewers, owner, contractor etc, at the design and review stage is valuable. Collaborative work allows the exchange of design review information for purposes such as notification and clarification, and processing information for monitoring, negotiating, and decision-making. Correct decisions can be made, and errors, conflicts or other inconsistencies can be detected early, thus preventing potential problems.

This cooperation ensures that everyone impacted by the design has early access to design information and the ability to influence the final design early, effectively and efficiently. A means for real-time collaboration within the same 3D space is thus needed to be included in the design review process.

1.3 Hypothesis

This research theorizes that design review process can be improved if valuable and presently scattered information is integrated and centralized. Information filtering based on context-aware concept coupled with the benefits of a VE can unify the presently "dispersed" and "scattered" design review information during design review. The use of context-aware concept allows for access of various design review information categories in real-time.

1.4 Objectives

The objective of this research is to develop a framework for context-aware information processing for design review in a VE. As such, the components and structure for information exchange and processing (access, retrieve and store) during a design review session is defined.

In achieving the objective, the following issues were explored:

- What filtering and query mechanisms are needed to retrieve the data/information during design review?
- What information should be displayed to the reviewers in the VE?
- What types of data/information should be assigned to each design objects (e.g. material type, part number, dimensions, building system type, building codes, project description, etc.)?
- How will reviewers' comments be input, retrieved, and stored by other reviewers?

Also, based on this objective, a work-in-progress (WIP) prototype design review application is developed. The purpose of developing the prototype was to demonstrate some of the proposed functionality of design review in a VE.

1.5 Research Contribution

The contribution of this research is a framework for context-aware information processing (access, retrieve and store) for design review in a VE. Only filtered and relevant design review information is presented for the respective design reviewer. This information is displayed with the 3D CAD model in a real-time interactive VE. This information is filtered and retrieved using context-aware concept. Four contexts of filtering design review information were drawn from content analysis of design review documents. The four contexts are:

1) discipline centric – based on the type/role of reviewer,
2) task centric – based on the task performed by the reviewer,
3) location centric – based on the location of the reviewer in the VE, and
4) object centric – based on the object selected by the reviewer in the VE

Figure 1-3 below shows the framework for context-aware information processing for design review in a VE which consists of the context-aware Information processing component and the VE component.

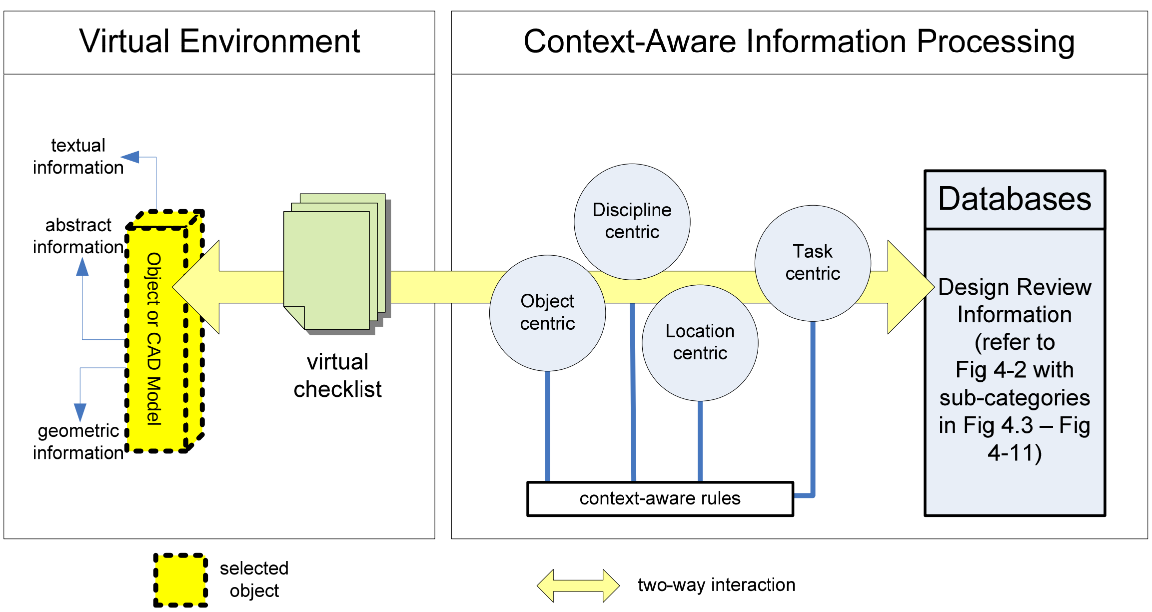


Figure 1-3 - A diagram showing the framework for context-aware information processing for design review in a VE

The context-aware information processing component is responsible for filtering, retrieving, and storing the design review information during design review. This component consists of a collection of databases where design review information is stored, and the context-aware rules that accommodate the information processing of the design review stored in the databases. The VE component consists of 3D Objects or CAD model, virtual checklists, and information in form of textual, abstract, and geometric. More details on the framework can be found in Chapter 4.

1.6 Research Methodology

The author used Action research style in conducting his research, designing and developing the framework. Figure 1-4 depicts the research methodology the author undertook. Following the Action research style, the author synthesized three methods which include review of literature, a case study and dissemination of his progressive work. The author used literature review to explore the current state of design review in the construction industry. The case study is divided into two sections i.e. interviews with industry personnel and content analysis of design review documents. Interviews were used to get an overview on the thoughts and opinions of relevant construction industry personnel concerning design review. Content analysis was used to analyze design review documents to identify and categorize design review information. At the same time and while the research work was progressing, the author disseminate his progressive findings in conferences, conference proceedings and journal publications.

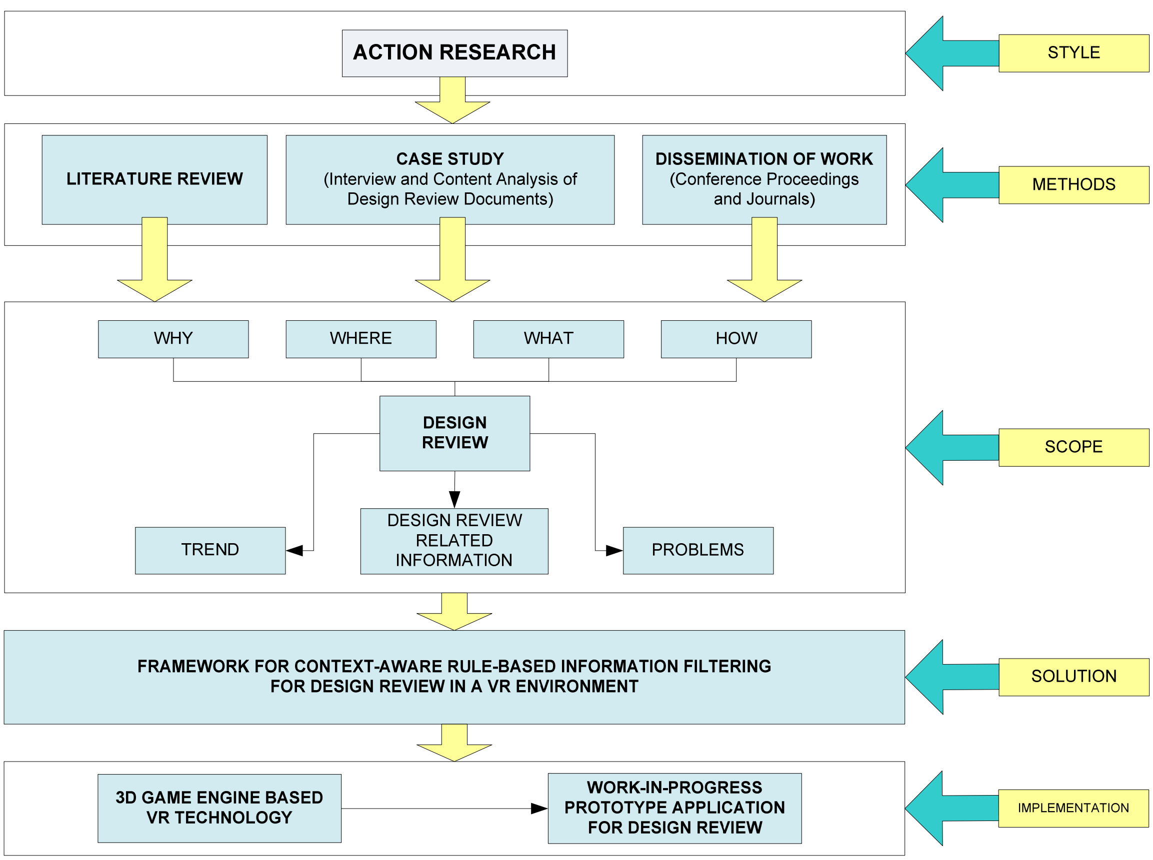


Figure 1-4 - Research style and methods undertaken by author for this research.

The author then developed the framework by using the findings from the synthesis of the three methods, and incorporated context-aware concepts for the processing of design review information into the framework. A work-in-progress prototype design review application in a VE was developed to demonstrate some of components of the framework. A 3D game engine was use as the enabling VE development tool.

1.7 Research Steps

In carrying out the research objectives, the following steps were taken (these steps are discussed in detail in Chapter 4).

I) Identification of the design review visualization needs (thus information to be displayed in the VE), and the generation of links among this information.

The main deliverable was identifying the main components and subcomponents for the design review information. Identification of the information components was carried out through the literature review of academic journals, white papers and technical reports. Information was also extracted from interviews with AEC industry personnel who are involved in design review, and from content analysis of design review documents from three Virginia Tech's past projects. The projects were:

1) The Career Services Facility Building
2) The Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech Complex
3) The Bioinformatics Phase 1 Building

Findings from the content analysis lay out the building blocks of design review information which is required for design review. This information include information and its attributes that are to be populated in the framework that serves design review needs in a VE, information and its attributes for 3D objects/model.

A context-aware concept with rules were developed to create links among the identified information above, and to provide a systematic way of visualizing information and performing design review activities in the VE. The context-aware approach supports four methods of filtering information in providing and displaying the relevant information to reviewers in the VE. These contexts are:

1) discipline centric – based on the type/role of reviewer in the VE
2) task centric – based on the task performed by the reviewer in the VE
3) location centric – based on the location of the reviewer in the VE
4) object centric – based on the object selected by the reviewer in the VE

II) Implementation and use of VE technologies, and suitable Graphical User Interface (GUI) design to support real-time interaction and manipulation of 3D geometric object and its related (spatial, perceptual and textual) information for design review in a VE.

Principles of 3D interaction, manipulation techniques and visualization of 3D object and information, and graphical user interface (GUI) design were reviewed. A low cost 3D Game Engine was used as the enabling VE technology to develop a work-in-progress prototype design review application.

III) Development of a work-in-progress prototype design review application in a VE.

The work-in-progress (WIP) prototype design review application is developed using a 3D Game Engine. The 3D Game Engine is used as the enabling software development tool for developing the VE. The 3D Game Engine used is the Torque Game Engine developed by GarageGames. The purpose of developing the prototype was to demonstrate some of the proposed functionality of design review in a VE.

1.8 Scope and Limitations

The research scope and limitations are as follows:

1) It is assumed the envisaged design review system will use a 3D CAD model that has been prepared by the designer throughout design phase. The 3D CAD model will then be reviewed by the design reviewer in the VE.

2) Implementation of prototype design review application is only limited to a desktop personal computer.

3) In this research, a prototype design review application in a VE was developed utilizing a 3D Game Engine. A hypothetical 3D CAD model of a 3 bedroom with 2 bathroom single-family home was used in the prototype design review application.

4) This research only addressed the classical design-bid-build project delivery method. Future research may address other types of project delivery methods.

5) This research neither directly addresses nor implemented procedures and schemas proposed by Building Information Modeling and Industry Foundation Class. Future research may consider developing data structure (3D CAD model and information) that is in compliance with BIM and IFC.

1.9 Industry Relevance

The introduction of a new design review approach where design review is performed in a VE will have the following potential benefits:

1) By using VE, many potential problems during construction as a result of design-related issues can be avoided. Design and design review team can visualize a 3D model of a facility in a real-time VE, and make decision to resolve any foreseeable problems that may occur.

2) The collaborative nature of a VE allows for design review to be performed in a collaborative virtual environment. Designers, design reviewers, and other project participants can communicate and collaborate during design review at the design stage. Communication and collaboration will be improved as the same 3D model is referred to and visualized in the same virtual environment.

3) The framework developed in this research allow for design review information to be held in a centralized location, instead of scattered in many places. The information can be visualized in the VE. The centralized information repository will assist design reviewers in making decision and perform design review more quickly. Through the context-aware rule-based approach the right information can be acquired by and displayed to the respective reviewer. In the current design review process, reviewers have little time to perform their review task more thoroughly. With a 3D real-time visualization tool and a centralized information repository in the VE, seeking information manually will be avoided. Therefore, design reviewers will have more time to perform design review more thoroughly in the same allotted time frame.

4) Using a 3D Game Engine can be a viable development alternative for AEC-related software developers. Many of the features exist in a 3D Game Engine can be fully harness for real-world application development such as design review software tool.

1.10 Document Organization

Chapter 1 provides a walkthrough of the research. These includes the introduction to the research, the problem statement, the hypothesis, objectives and research contribution, the research steps, the research scope and limitations, and relevance of the research to the construction industry.

Chapter 2 reviews literature on design review that includes the current practices of design review, the current drawbacks, examples of design review, the benefits of performing design review and the information used during design review. This chapter also reviews Virtual Reality, its characteristics, some examples on the use of VE in the construction industry, and the visualization of information in a VE. Since the author uses a 3D Game Engine as the enabling VE development tool, a review on 3D Game Engine is also included, with discussions on the strengths of the engine.

Chapter 3 describes the gathering and analysis of author's research data. The research style the author undertook was Action Research style. This involved the synthesis of methods including Literature Review, Case Study (Content Analysis of design review documents and Interviews) and, dissemination of his work-in-progress in conferences, proceedings, and journals.

Chapter 4 describes the framework for the context-aware information processing for design review in a VE.

Chapter 5 concludes the research. It provides an overview of the design review framework, the author's motivation of undertaking the research, the summary of how the research was established, the limitation of the research, and finally, potential future works in the same research area and the research direction that can be undertaken.

CHAPTER 2

2 LITERATURE REVIEW

2.1 Introduction

This chapter reviews literature on design review such as the common practices of design review in construction and their weaknesses, the benefits of performing design review, and the trends and types of design review information. This is followed by a literature review on Virtual Environment (VE). The areas reviewed are the definitions of VE, the characteristics of VE, and visualization of information in VE. Finally, this chapter reviews 3D Game Engine as a tool for developing VEs. A general review of 3D Game Engine and its strengths are described.

2.2 Design Review

The main objective of design review in many industries is to make sure the project's priorities are met. Design review makes up an essential part of modern industrial practice. If design review is properly performed, it provides a mechanism whereby the total design activity can be carried out in an objective manner, leading to improved designs and products (Pugh, 1991). The term "design review" may cause confusion because different disciplines may interpret the term differently. The types of design being reviewed has to be first be defined. The word "design" is used commonly in many circles but describing different situations. Engineers use the term to refer to the technical synthesis where a concept moves from an idea to a functioning product or system. Graphic artists also use the term to point out a creative process, but this use tends to be artistic and aesthetic rather than technical. In the artistic instance, the term design may not involve any functional aspects. On the contrary, engineers are used to think of the term design as functional requirements. Further, the term can be used to refer to both a process and a resulting product, for example "I am going to design a house" or "The design is complete". In this research, "design" is used in the engineering and technical context. The term "design review" in this research refers to the reviewing of completed drawings (depending on the stages of design; programming, schematic, preliminary and working drawings) which are prepared by the designer/architect/engineer.

According to Ichida (1996), "Design review is not about criticizing the work of design engineers and making lists of faults and errors. It is a cooperative undertaking in which people from other areas contribute their expertise to produce better designs from the beginning". Design review is not intended to find errors or incompetence of the designer in creating design. Specifically in the construction industry, the design reviewer's role is to periodically check the design and design documents for inconsistencies, errors and other aspects of design which does not concur with the owner's requirement (East, 1998).

2.2.1 Design Review in Construction

In the construction industry, design review is a process in which information is exchanged by the designer and reviewers to resolve and achieve the interests, goals, and objectives of the owner in a timely manner. A design review process is used by design and construction sector to help ensure the quality of designs (East, 1998). The design review process reviews the materials to use; the building systems components (structural, mechanical, electrical, plumbing etc.); compliance with the current codes and standards; and the maintenance requirements. Work by East *et al.* (1995), East (1998), Spillinger (2000), and East *et al.* (2004) reveal design review is crucial for detecting and identifying conflicts, errors and inconsistencies in designs

Typically, a construction design review process is an iterative activity; carried out in four consecutive stages at the end of each design phase (Spillinger 2000; East *et al.* 1995; Fu & East 1999). These stages are Programming, Schematic, Preliminary and Working Drawings. 2D CAD drawings either digital or paper-based and specifications are the documents that are heavily used during design review. These documents are prepared by various specialists such as the architect, electrical engineer, structural engineer, mechanical engineer, etc. Thus, coordinated and error-free design documents are difficult to achieve. For example; a designer produces designs in 2D format (hand or computer-aided) which is derived from his 3D mental models in their internal and personal mental space and converted into external 2D designs and information on paper. To interpret the designs especially for review purposes, others need to have the skill to transform 2D designs into 3D spatial models or structure, solely based on their own experience and interpretation (Dunston *et al.*, 2003). The process of interpreting the designs is undoubtedly influenced by various individual standards, resulting in the opportunities for errors and conflicts in design communication, and during construction. According to Nigro and Nigro (1992), errors and omissions in building design typically account for half of a project’s change orders. The remainders are normally because of various unforeseen and/or inconsistent site conditions, and changes in the owner’s needs. On average, there are usually five errors or omissions per contract drawing. Therefore, on a project with 100 drawings, there are potentially 500 problems (Redicheck, 2003). With various disciplines producing drawings for most projects, keeping the drawings coordinated is essential to avoid errors and omissions.

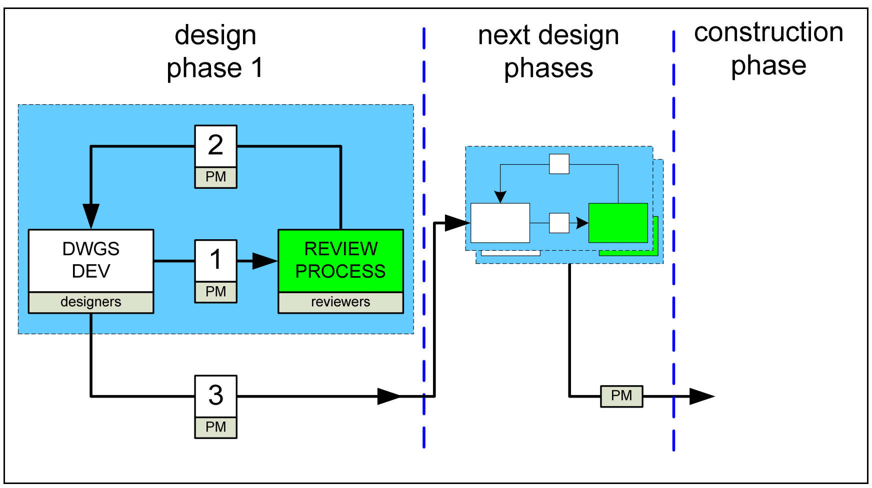
Potter (1989) believes that “Drawings are a means to the end for the recipient, their expressive content being strictly limited to conveying instructions; they are not the end product in the process”. This is where the role of the reviewer comes in. Reviewers ensure that drawings are as accurate as possible, free from error, as the means of communicating information between all members of the building team, and will not cause confusion to the building team and the users of the designs. Well-crafted, visually impressive drawings are of little use unless the information they convey is correct (Emmitt, 2002). Huang (2002) suggested that other than the designer and the reviewer, an important key player in the design review process is the Project Manager or Review Coordinator. He is the point of communication and responsible for the channel of communication between the designer and the reviewer, and the design review team with the approval bodies, and the approval bodies and regulators with the project.

2.2.2 Current Design Review Practices

East *et al.* (1995) and Fu and East (1999) described the typical process for conducting design reviews. The process includes three steps:

1) the reviewer examines the plans and specifications for a project
2) the reviewer, drawing information from experience, standard references, or from other advisers, notes potential design errors or omissions found
3) the reviewer writes a list of review comments to be not only returned to the designer, but also shared with other reviewers

Figure 2-1 below shows that drawings as the input of design review are passed from the A/E to the reviewers [node-1] for comments on any inconsistencies found. Once the comments are made by the reviewers, the drawings are given back to the A/E [node-2] for redesign and for incorporating the required changes. The process of design, review and redesign is cyclical in nature until satisfactory drawings [node-3] are accomplished and approved by an Approval Body. Once approved, the drawings for that design phase are considered complete and the A/E can start with the consequent next design phase and finally prepare for construction. The project manager or design review coordinator ensures the right flow of documents and communication among the parties involved. This process is tedious and intensive because of time and resource constraints imposed on the reviewers (East *et al.* 1995; Ichida 1996; Fu & East 1999; Staub-French & Fischer 2001).



Note: PM - Project Manager
DWG - Drawings

Figure 2-1 - The cyclical nature of drawings/design development and the design review process

Design review can be in the form of designer's (self-checking) review, peer (in-house) review, and external reviews by the owner, independent designers, or contractors (Andi & Minato, 2003). Several design review practices are commonly used by the construction industry. Design review is performed using checklist (paper-based and/or computer databases). Checklist is a mean to systematically guide the reviewer throughout the review process (for a sample checklist, refer to Appendix 1). Design review related information may be available from internal and external databases of similar past projects. The use of a `Light Table' is a typical practice. This is performed by overlaying each specialty designers' 2D drawing on a light table to compare the different building system designs. The designers identify conflicts and develop new solutions. This process continues until the coordination is complete and the specialty designers sign-off on each other's drawings to show their acceptance.

The use of a life-sized physical mock-up (PMU) (Figure 2-2) of a building or facility to be built is used by large and resourceful companies (Staub-French & Fischer, 2001) such as Pulte Homes, Inc. (Shiratuddin & Thabet 2003a; Shiratuddin & Thabet 2003b; Cowden *et al.* 2003). Since many of the project participants are geographically spread, they schedule to travel to attend the design review session at an agreed location. During the design review session, the project participants inspect and discuss various aspects and components of the design by physically walking through the PMU. The outcome of every discussion is written down, or drawn directly onto the 2D drawing, design errors are noted, or improvements to the design are suggested. Decision is then made whether the model must be reworked, in need of further improvement or the model is a perfect prototype and ready for the final massive construction. If changes are required, then modified information are given back to the designer to incorporate the new changes made onto the 2D drawings.

Figure 2-2 - An example of a physical mock-up used for a design review session (Courtesy of Pulte Homes, Inc.)

2.2.3 Drawbacks of Current Design Review Practices

According to East (1998) design review is still imperfect despite the current available tools because reviewers are subject to many pressures in the design review process. East first highlighted that design review is inherently a time-consuming process. Second, review is a resource-constrained process; assigning experts to "more critical" design and construction tasks create resource bottlenecks. Third, time constraints because of backlogs of unreviewed drawings and specifications may force reviewers to sacrifice the thoroughness of their reviews. Also, time constraints on project funding cycles constrain the ability of the designer to incorporate changes that have been recommended by design reviewers. As a result, jobs may be bid with known mistakes. East believed the potential of design review is promising but current methods are still imperfect thus causing the benefits to be unattainable.

The current methods of design review lack a central and integrated information base to make the flow of information more efficient. Information is still at large and spread across at various places. The current process of overlaying 2D drawings on a light table is time-consuming and inefficient (Staub-French & Fischer, 2001). Many design conflicts go undetected as 2D drawings do not adequately represent the spatial information and requirements of the building components. Using the light table and the interdisciplinary checklist are purely manual, relies heavily on paper-based formats, and require the use of manual cross-checking of drawings and documentations seeking for the required information. The method of using web-based online review system, involves perusing web pages, which is to some extent comparable to paper-based format, only substituted electronically by the computer and have some information storage and retrieval capabilities. The method of using PMUs is expensive and time-consuming to both the project and team members, and lack of efficient way of handling information centrally and efficiently.

According to Christiansson *et al.* (2000), current software tools supporting detailed design review are already available. Nevertheless, these existing tools suffer from two important limitations. First, they lack of 3D-real time inspection features. Second, the discontinuities or non-inter-operability between the different software tools makes the reuse of the results of one phase in the design process as an input for another phase impossible.

2.2.4 Examples of Design Review System in Construction

a) RediCheck

In 1981, William T. Nigro developed Redicheck review system for design review (redicheck-review.com, 2006). It is a system specifically designed to address points of interface, allowing production personnel or a quality assurance reviewer to establish coordination discrepancies between disciplines and locate design coordination errors and omissions. Redicheck is both a systematic method of construction document quality control and a professional group of Architects and Engineers who are authorized to perform Redicheck services directly for owners. The Redicheck system uses an interdisciplinary coordination paper-based checklist and overlay checking process. The interdisciplinary checklist is based on the sequence of construction i.e. civil checklist, structural checklist, architectural checklist, mechanical/plumbing checklist, electrical checklist, kitchen/dietary checklist, specification checklist and finally a consolidated Redicheck checklist. All the Redicheck checklist items are coordinated by 60% completion of the construction document production stage. As each item is checked, problem or discrepancies with other disciplines is communicated. The disciplines where the solutions may affect are also notified.

b) Reviewer's Assistant and DrChecks

The Reviewer's Assistant is a design review tool developed by East *et al.* (1995) for the U.S. Army Corp of Engineers. The Reviewer's Assistant assists reviewers by capturing, storing, and retrieving design-review comments, and compiling lessons learned. The Army Corps further developed the DrChecks/Construction Lessons Learned (CLL) software program (East *et al.*, 2004) which is used to conduct project reviews by recording review comments and designer annotations. Once the comments and/or annotations are recorded, others can instantly view and respond to them over the Web.

In 1998, Design Review and Checking System DrChecks were developed. The three key design considerations for developing DrChecks are shown below in Table 2-1. The U.S. Department of State's Overseas Building Operations (OBO) use DrChecks as the official tool to support the integrated review process used at the State Department and has provided a structure for the review process (East, 1998).

Characteristics necessary for development of DrChecks	
Improving execution of design review:	• The system must be readily available, fully supported, and well documented • The system must support electronic submission of comments to project managers • Reviewers must be able to read other reviewers' comments and follow discussions • Comments must be able to be sorted by disciplines • Designers must be able to answer comments directly to • reviewers and resolve issues before the review conference • Supervisors must have the ability to screen/scrub reviewer comments before they are submitted to designers
Improving management of the design review process:	• The system must support real-time review status reporting • The system must foster the ability to communicate between participants
Support the capture and reuse of corporate knowledge:	• Prior review comments must be easily retrieved and reused • The system must allow capture, validation, and reuse of approved lessons learned while the application is being used

Table 2-1 - Characteristics for development of DrChecks (Source: East, 1998)

c) iReview

iReview is an online review and comment service (http://www.ireview.com). As with DrChecks, iReview allows scattered reviewers to access, view and track project's documents and reports, place comments and respond to variety of topics. iReview provides a full range of features designed to address the challenges of communication and coordination between multidisciplinary teams during the design phase of the construction process.

2.2.5 The Benefits of Design Review

The design review process produces immediate benefits (East *et al.* 2004; Ichida 1996). First, the process avoids future contingencies by detecting design errors, omissions, and inconsistencies before the project goes out for bid. As a result, future claims and/or future delays are avoided. Second, the design review process improves the quality of the final facility and the overall project. Building or facilities are more functional, have a reduced environmental impact and are safer for the occupants. Third, a design review system that increases productivity will not only reduce the cost of carrying out the design review by reducing the number of participants and the time necessary to complete the review phases, but will also shorten the facility delivery cycle. Reduced delivery duration will provide the owner with favorable occupancy earlier. Earlier occupancy translates into significant total project savings to the owner.

Soibelman *et al.* (2003) refer to the report by Spillinger for the Federal Facilities Council (FCC) in 2000 which pointed out what a good design review should be able to do:

- Detect problems related to errors, omissions, and inconsistencies caused by design process;
- Identify inappropriate construction methods and materials that would increase construction costs;
- Certify compliance with standards, specifications, codes, and regulations;
- Identify components and building systems that would result in making facilities difficult to maintain and operate;
- Increase project quality, integration, bid-ability, constructability, and maintainability;
- Reduce construction change-orders with consequent decline in construction delays associated with them;
- Decrease number of requests for information (RFI) during construction;
- Gain knowledge that can be applied in other phases of project life cycle, which can improve returns gained from this knowledge; and
- Improve owner satisfaction and usefulness of the facility.

A study for the FFC shows that total project cost may be significantly reduced by conducting an effective design review process (Spillinger, 2000). This study found that effective design review processes add value by saving time and money over the entire design and construction process. It was strongly suggested the team responsible for design review should include representatives of all project stakeholders that usually includes the owner, the user, the A/E, the main contractor, operation and maintenance staff, and major equipment vendors. Areas of savings that can be achieved by effective design review as suggested by Spillinger (2000) in his report for the Federal Facility Council (FFC) include:

1) less rework on the part of the construction contractor

2) fewer change orders to the owner for correction of design errors or omissions and the cost of late project feature upgrade that should have been addressed in the original design, and
3) avoiding costs associated with loss of productivity during construction-delayed facility start-up, and litigation

Spillinger concluded that effective review of designs maximizes the probability that a business requirement will be successfully supported by a facility that was conceived, designed, constructed, and placed into operation efficiently and effectively.

Redicheck design review system that was developed in 1981 by William T. Nigro reports three case studies on cost savings. One study was on the construction program at the Naval Trident Missile submarine Base, Kings Bay, Georgia (RediCheck Associate, 2007). The total value of a construction for the study was $670 million which included many separate construction projects from 1979 to 1986. Most of the projects were competitive sealed bid designed by various A/E firms. A total of $250 million of construction was performed before RediCheck was implemented in 1982. A total of $420 million of construction was performed after the RediCheck System was adopted. There were no significant changes in design, construction, or administration of projects were implemented. Change order rates and change order types were tracked from the beginning.

After using the RediCheck System the project change order rate from 1979 to 1986 dropped, from an average 10% to 3% of the total construction value. This reduced the total program cost of $420 million by roughly 7% of the construction value reviewed. Roughly one half of the change orders were directly attributed to design errors or omissions. The study further reported the total estimated cost savings to the Navy because of RediCheck adoption were estimated to be 3.5% of construction cost. This was a savings of nearly $15 million (3.5% X $420 million = $14.7 million).

East *et al*. (2004) describes a cause-effect diagram of the benefits gained from using a design review process in construction. Design review process produces three immediate benefits. The first benefit (effect) is that future claims and/or future delays are avoided. This is caused by avoiding future contingencies via the design review process which detects design errors, omissions, and inconsistencies in design and design documents before the project goes out for bid. The second benefit (effect) is the much improvement on the quality of the final facility and/or civil works project because of the design review process. Those improvements translate into facilities that are more functional, have a reduced environmental impact, and are safer for the occupants. The third benefit (effect) is increased productivity which not only reduces the cost of design review (by reducing the number of participants and time required to complete the review) but also shorten the delivery of the facility. Reduced delivery duration will provide the customer with favorable occupancy earlier. Earlier occupancy translates into significant total project savings to the customer.

Harvey (2006) reports a successful case study on the use of their design review system, Autodesk Design Review (ADR). Crom is a firm which designs and builds prestressed concrete water storage tanks for owners across the southeastern United States and the Caribbean. Crom used the ADR system with AutoCAD which they have already been using. It is reported the drafters spend more time creating designs in AutoCAD and less time on the design review process. When it was time to start the design review process, a drafter simply publishes DWFTM files of shop drawings directly within the AutoCAD application, and the project manager sends them out to reviewers by e-mail. Each reviewer opens the DWF in ADR, uses the markup tools to make changes, and passes the DWF to the next reviewer. ADR automatically tracks who made changes and when. After the process is complete, the drafter efficiently incorporates changes into the AutoCAD drawing. ADR makes easy Crom's workflow, including eliminating time-consuming faxes, lower delivery costs, and a reduced risk of errors. It is also reported that productivity improvements is experienced mostly by the firm's AutoCAD users. With ADR only one version of design is used for review, instead of three before using ADR. Crom designers spent 10 to 20 percent less time on design review.

2.2.6 Design Review Information

In construction, information is required for many different purposes, for developing design, approvals, for building and for record purposes. For example, the project managers are constantly flooded and rushed by requests for forms, project information and documents, perhaps even contract disputes and litigation (Lam & Chang, 2002). Information generation becomes more intensive as design stage progresses. The use of valuable information during this stage can influence the quality and the total cost of a facility (Lutz, 1990; East *et al.* 1995; East, 1998; Spillinger, 2000; East *et al.* 2004). Thus, the large volume of incoming information and documents, and how to effectively respond to them needs to be systematically organized. Spillinger's (2000) report for the FFC stated that 30 to 50% of all construction change orders result from errors in the design document that are directly related to improper interfaces between design disciplines. Many faults in buildings can be traced back to incomplete and inaccurate information, and the inability to use information that has been provided (Lutz, 1990). Discrepancies between drawings and specifications can and do lead to conflict. Often relevant information may not be available when needed leading to a constant reinventing of the wheel (Veshosky, 1998). Even if information is available and shared, the meaning of the message may not be correctly interpreted or understood (Pietroforte, 1997). Emmitt and Gorce believed that some of the problems can be avoided with well-designed systems providing useful information and information coordination.

Elsewhere, a survey conducted by Baldwin *et al.* (1999) highlighted some information-related problems during construction design processes. Some of the issues distinguished by the survey participants were withholding information among design team members against one another. This leads to loophole and missing information, and incomplete circulation of information. Eventually, the poor quality of exchanged information leads to poor judgments and assumptions. Since multiple organizations and individuals are involved at different phases; therefore, no one can claim to have all the most up-to-date information all the time.

Kubicki *et al.* (2006) researched the building construction team interactions, exchanges, and communication in building projects. They identified the various people involved and the varied sources of information shaping the current methods of interactions, exchanges, and communication did not allow for creating relations between information and the easy tracing of events. Occasionally, information is shared in its totality and to people who is not concerned. Information is exchanged in various formats, some paper-based and some digital, is not linked, thus searching for an exact piece of information is difficult. Finally, these methods are not easily adaptable to the changes in the project. Refreshing and updating of information to represent the building construction progress is difficult. These problems penalize the demonstration of activity and therefore resulted to interruptions to the development of the project in forms of delays, mistakes, and defective works. Both Lam and Chang (2002), and Baldwin *et al.* (1999) have the same argument that an effective information and data management system, with an efficient database, is a sensible solution to the problem of information handling and management in construction. By understanding the information and the needed processes, the management of the design process can be significantly improved, thus leading to better design and construction.

The Corps of Engineers recognized the need to develop an automated review management system capable of supporting the design review process as early as the mid-1980s (East *et al.*, 2004; Lutz *et al.*, 1990). The focus of the effort was to develop a mainframe system to support the collection, collation, and exchange of information required to perform the review process. The initial experience, although partially successful, was limited by remote connectivity to the mainframe application. The rapid growth of the World Wide Web has prompted a review and update of the concepts of remote collaboration on design review (Fu & East, 1999).

Fu and East (1999) discussed the handling of information for viewing, retrieving, and storing is a significant component in introducing a new approach to improve the way design reviewers work. Handling information, involves improving how information, typically paper documents in the current design review process, is viewed, retrieved, and stored. The drawings and specifications given to the reviewer in the current design review process are paper-based, two-dimensional, and textual. Often the reviewer needs to cross-reference drawing sheets and the specification documents to visualize necessary information. In Fu and East's (1999) virtual design review, reviewers were able to index and to view both a three-dimensional and a traditional plan of a building. Design reviewer's output from a design review process is the generation of review comments. These comments can be organized to allow quick access so users may see the comments created previously. Organizing the comments require that users provide indexing information for every comment submitted such as specification section, drawing sheet number, applicable design discipline, at least two keywords, phase of review, category of problem, and major building system

Information as part of communication is an integral part of any construction effort. In 2000, Kuprenas suggested the future trend of communication characteristics of the AEC industry holds certain promises, however, until today, the AEC industry have still a long way to go. Some of the trends Kuprenas suggested were:

1) the integration of development of design by all project parties, however the study of organizational alternatives for new production methods, the communication needs and how to fulfill them needs to be identified first
2) the open and immediate access to all project information by all parties in all phases. This can be achieved through the development of a single standard to store all types of data, development of practical methods to retrieve and present all types of data

3) design and construction process to be guided by virtual smart models
4) there should not exist division between project participants
5) emphasis should be put on the program rather than the project

The ideal situation envisaged by Kuprenas is hard to be achieved as argued by Akinsola *et al.* (2000) and Bouchlaghem *et al.* (2004) mainly because of the fragmented nature of the industry. In general, construction processes involve many participants where most operations and tasks contain basic work processes. These processes are supported by systems that are isolated. Different professions also use their own unique processes to carry out these operations and tasks. Changes, whether minor or major to the way of working will need to be introduced to achieve integration.

2.3 Virtual Reality

The term Virtual Reality (VE) was first coined by Jaron Lanier in 1987. Lanier's work in research and engineering contributed several products to the VE industry (Virtual Reality 2007; Behr 2002). The seeds for VE however were planted in several computing fields starting in the 1950s. One of them was the creation of a prototype by, Ivan Sutherland in the early 1960s (Sutherland, 1963). Sutherland prototyped the first graphical computer system called Sketchpad, which by using a light pen, drawing of vector lines can be made on a computer screen. This creation has influenced the way every computer user thinks about computing. It has made fundamental contributions in human computer interaction and as one of the first Graphical User Interfaces (GUI). This breakthrough by Sutherland was followed by two years later when he described the concept of the Head Mounted Display (HMD) and an immersive 3D computer environment in his paper "The Ultimate Display" (Sutherland, 1965). This invention has made Sutherland a celebrated contributor to the technological artifact most often identified with VE. The concept of VE has led to much research and exploration of its use, and followed by applications and implementation that benefits society.

VE has been variedly defined by academic circles and industries. To some people, VE is a specific collection of technologies. While others stretch the term to any medium that can present an environment that draws the receiver into its world, which includes conventional books, motion pictures, radio, and so on (Isdale, 1998). Lok (2004) however sees VE as immersive VE that are made of systems that allow participants to experience interactive computer generated worlds from a first-person perspective, as opposed to prerendered movies, videos, or animations. Lok's opinion matches to Foley's (1987) which is "the goal of VE is to place the user in a three-dimensional (3D) environment that can be directly manipulated, so the user perceives interaction with the environment rather than the computer". Fred Brooks, another notable pioneer of VE, defines a virtual reality experience as any situation in which the user is effectively immersed in a responsive virtual world, and the user have a dynamic control of his/her viewpoint (Brooks, 1999). Other definitions of VE are:

> "…a way for humans to visualize, manipulate and interact with computers and extremely complex data." Aukstakalnis and Blatner (1992).

> "…an experience in which a person is surrounded by a three-dimensional computer-generated representation, and is able to move around in the virtual world and see it from different angles, to reach into it, grab it, and reshape it" Rheingold (1991).

> "…the representation of a computer model or database, which can be interactively experienced and manipulated by the VE participant(s)." (Barfield & Furness, 1995).

> "…the use of computers and human-computer interfaces to create the effect of a three-dimensional world containing interactive objects with a strong sense of three-dimensional presence." (Bryson, 1996).

2.3.1 Characteristics of Virtual Environment

Burdea and Coiffet (2003) noted that VE as comprising of three main characteristics: "The three I's of virtual reality: *immersion, interaction*, and *imagination*". While according to Sheridan (1992) and (Slater *et al.*, 1998), *presence* is also an important characteristic whereby VE draws out the feel of presence or *being there*. With the sense of presence, users are physically immersed in the VE, and receive sensory information and experience similar (or almost similar) to the real-world.

2.3.1.1 Immersion

Immersion is a feature essential to a VE (Burdea & Coiffet 2003; Pinho *et al.* 2002). The Merriam-Webster dictionary defines immersion as "the state of being absorbed or deeply involved" (Webster, 2006). Witmer and Singer (2002) described immersion as a state where one perceives oneself as being enveloped by, included in, and interacting with an environment that provides a continuous stream of stimuli and experiences. Witmer and Singer elaborated that these experiences can also come from simulation rides, IMAX theaters, and to a lesser extent standard movie theaters that can be said to "immerse" one's senses.

An immersive virtual reality environment can be experienced in a standard Computer-Aided Virtual Environment or CAVE™ for short. A CAVE™ is cubicle in shape and can have between four to six screens placed on each side. A four-sided screen CAVE™ configuration will usually have screen placed in front, left, right and bottom. A fully enclosed CAVE™ display will have all six-sided screens. The surrounding projection screens allow users to be fully immersed (Figure 2-3a). Another example of immersion is when a user wears a head-mounted display (HMD) (Figure 2-3b), or standing inside a spatially immersive display (SID) or sometimes known as Immersive Workbench (Figure 2-3c).

a) The CAVE™ system at Virginia Tech | b) The author wearing a Head-Mounted Display | c) Immersive workbench at Virginia Tech

Figure 2-3 - Types of immersive VE displays

Immersive VE systems use 3D interaction techniques based on whole-body input (Poupyrev *et al.*, 1996). For example, to see what is to one's left, he turns his head/body to the left; to move forward he takes a step forward; to grab an object he reaches out his hand and grab the object with his fingers. These natural techniques can be improved for greater efficiency or usability, such as one can use a virtual extended arm in the Go-Go technique (Poupyrev *et al.*, 1996) to reach much further than one's normal and physical arm.

2.3.1.2 Interaction

VE interfaces usually support both explicit and implicit style interactions (Bowman & Hodges 1997; Pouperev *et al.*, 1996). Conventional interfaces support mainly explicit style interactions such as the use of a keyboard and mouse to navigate or point to objects in a VE. Implicit style interactions allow more natural and easier to use human-computer interactions by allowing arm, hand, head, or eye movement based interactions. Implicit style interactions are more complex to design compared to explicit style interactions. Witmer and Singer (1994) suggest the greater the level of control and interaction a user has in a VE, the higher the reported level of presence. Witmer and Singer explain that this control factor is driven by the immediacy of the systems response, corresponding user-initiated actions, and by the naturalness of the mode of control. Parallel to Burdea and Coiffet's (2003) interpretation of VE as "a high-end user-computer interface that involves real-time simulation and interactions through multiple sensorial channels", VE allows a user to be actively involved. A user can move by walking in or fly through in a virtual model, with unconstrained movement, i.e., no predefined path exists (Shiratuddin & Thabet, 2002; Shiratuddin & Thabet, 2003).

Interaction in a VE is provided through various spatial input devices (Lok 2004; Pinho *et al.*, 2002; Bowman *et al.*, 2001), most providing at least three degrees of freedom (DOF) based on the tracker technology used. Such devices include 3D mice, various kinds of wands with buttons for pointing and selecting, data gloves that sense joint angles, and pinch gloves that sense contacts. With these devices, natural and thus quicker exploration of 3D models in the environment can be made. Body-centric judgments about 3D spatial relations also come more easily (Pausch *et al.*, 1997), as can recognition and understanding of 3D structures (Bryson, 1996). A user can move through the VE as desired; inspecting virtual objects from different positions, angles, and orientations.

2.3.1.3 Presence

Presence is "the perceptual illusion of non-mediation involves continuous [‘real-time’] responses of the human sensory, cognitive and affective processing systems" Lombard and Ditton (1997). It is the state of “being there” which is the degree to which the users feel that they are somewhere other than they physically are while experiencing a computer generated simulation (Slater *et al.*, 1998). Feeling presence is dependent on technological factors and individual factors. Banarjee *et al.* (2002) state that in VE devices, presence refers to the quality of the experience of being present in a computer generated environment and experiencing the “real thing”. Further, Banarjee *et al.* noted that different forms of VE devices or media can produce different levels of presence for the same individual. In addition, the same media can also produce different levels of presence in different individuals.

There are three types of presence: environmental, social and personal (Heeter, 1992). Heeter describes environmental presence as the extent to which the VE recognizes one’s existence and reacts to him/her. Social presence can be explained by whether multiple people exist/immersed in same VE, with the presence of others, these people can further believe the ‘existence’ of the VE, can experience higher levels of presence. Personal presence is a measure of the extent to which one feels like they are in the VE.

2.3.2 Virtual Reality in Construction

Burdea and Coiffet (2003) describe VE as “a high-end user-computer interface that involves real-time simulation and interactions through multiple sensorial channels”, and further highlighted the three I’s of VE: *immersion*, *interaction*, and *imagination*. Sheridan (1992) adds the characteristic of presence, and much research and work add the visualization criterion. These interpretations have been implemented in many VE applications in research labs and commercial products, but the use is limited to large companies and has not widespread into the general construction community.

In a 3D VE, construction project stakeholders can “climb inside” a building; visualize its elements and components from any visual perspective (Thabet *et al.*, 2002). By walking through a 3D model in a VE, designers can view and test the design directly (Ormerod & Aouad, 1997). One can also virtually “disassemble” and “reassemble” the components of a building, rehearse the construction process, develop a construction sequence, assess the constructability of the design, and identify potential interference problems. Unlimited virtual walkthroughs of the building can be taken to experience in near-reality sense, what to expect when construction is complete

Neville (1998) suggests that training is important for rehearsal purposes and to prevent accidents and injuries. Barsoum *et al.* (1996) developed an interactive virtual training model, “SAfety in construction using Virtual Reality" (SAVE), to train construction workers on avoiding falls from platform-metal scaffolding. Using HMDs, users are able to interact with the VE and detect dangerous conditions (e.g. missing guardrails, loose, weak or inadequately spaced planks, inadequate connections between scaffolding components, and defective components) and try to eliminate it. A scoring system is used to evaluate performance of participants. SAVE comprised of two main modules; an erect module, and an inspection module. The erection module is used to show proper procedures to erect scaffolding. The inspection module is used to detect and correct the potential causes of falls. Sense8 WTK on an SGI platform and 3D Studio were used for creating the VE and engine. An Onyx Reality Engine 2 (ORE2) from SGI was used to allow real time rendering. Soedarmono *et al.* (1996) also developed a prototype VE model for training personnel on avoiding falls during construction. Occupational Safety and Health Administration (OSHA) regulations were integrated into the model as 2D text and audio information. Warning messages with the required safety standards are displayed or announced when a user approaches a working platform in the VE from which they could fall.

A group of researcher from the University of Teesside, UK (Dawood *et al.*, 2005) has developed the Virtual Construction Site (VIRCON), a prototype application for evaluation, visualization, and optimization of construction schedules within a VE. The structure of the VIRCON system is designed with three main components: project database, analysis tools and decision support components. The VIRCON was found to be practical because it was based on the industrial requirements, real-life project data, and finally evaluated by the industrial collaborators. The industrial evaluation has proved that VIRCON is practical and communicative, and have potential towards commercialization and real implementation.

Einstein (2007) reports on Bechtel's suite of computer software programs that it uses to provide its customers with customized master planning and design solutions through VE. The software can simulate airfield, terminals, roads, and other airport systems. The software also uses animation and walkthroughs to show in advance what completed projects look like and how they work. VE walkthroughs were used to speed up the design process, confirm engineering decisions, and to some extent reveal design flaws that can be fixed before construction begins. Bechtel uses VE in its projects such as at the New Doha International Airport in Qatar, the new Tacoma Narrows Bridge in Washington State, and the Chernobyl power plant in Ukraine (Berkoe, 2005). A Bechtel-led consortium created a virtual model for the visualization of the model. It showed how the permanent shell would be built, and how the temporary shell that has covered the reactor for twenty years would then be disassembled and removed.

2.3.3 Virtual Reality and Visualization of Information

What a user see or view in a VE is important to create the connection between the user and the VE. As discussed earlier in this chapter, to term a system as supporting virtual reality, it has to have the *immersion, interaction* and *presence* characteristics (Burdea & Coiffet, 2003). These characteristics can only be experienced with a good interface design of the VE system. This interface is called the *user interface* or *human interface* and in a virtual reality system, many of the user interface will be visual or graphical, therefore the word Graphical User Interface or GUI. Mandel (1997) cited John Anderson's (1989) "The way a user interacts with a computer is as important as the computation itself; in other words the human interface, as it has come to be called, is as fundamental to computing as any processor configuration, operating system, or programming environment."

Design of user interface is important so users can accept and recognize what is presented to them. A system with outstanding features will not be successful if the user interface is not well designed. A well-designed user interface is important to users because it is the users' window to view the capabilities of the system. It is also one of the visible components of a developer's product (Galitz, 1997), ranging from a simple software program for a personal computer to a complex simulation program for a CAVE system. Galitz further explains that a user interface is also the vehicle through which many critical tasks are presented. Also, a user interface layout and appearance do affect a person, such as if the system makes them confuse and inefficient, he/she will have a greater difficulty in doing tasks and will make more mistakes. This may even chase or hinder some people away from the system permanently. A good user interface design is fundamental to a successful visual system such as a VE system.

Mandel (1997) describe user interface as "the whole experience between the user and the computer. It includes both the computer hardware and software that *presents* information to users and allows users to interact with the information and the computer." The hardware component is the physical element that makes up the system such as a keyboard, a mouse, a central processing unit (CPU), and the monitor as the display screen. The software component is all the items users see, hear, point to, or touch on screen to interact with the computer system, as well as the information with which the users work. It needs a careful crafting and expertise by computer designers and engineers for these components to interact with one another, to allow the presentation of information to the users, and to allow for users to interact with the computer. Mandel accounts on three areas (also known as the Mandel's Golden Rules of Interface Design) of user interface design principles which can be summarized as:

- Place users in control of the interface: The user is free to go where they want to go and how they want to get there. E.g. user can work on a spreadsheet, switch to a web-browser to cross-reference information on the internet, and at the same time listen to audio music
- Reduce users' memory load: The computer can assist the user to remember and carry out tasks that are repetitive. E.g. defaults such as, undo, redo, copy, and the use of icons
- Make the user interface consistent: Consistency is a key facet of a usable interface where users reuse their knowledge and what they learn to a new program. This can only happen if the programs are similar to what they have already used

An important principle of GUI designs is to provide users with the tools and applications to do a task. GUI designs should also be concern with the ability for users to directly manipulate objects and information to do the task on-screen or interface presented to them (Mandel 1997; Galitz 1997). The first factor to consider in presenting the information to the users is the amount of information the GUI should present at any one time. Helander (1988) highlighted that information clarity and readability is improved if information on-screen is showed in a less crowded fashion. The design of the interface should display only the information the user needs to perform his/her task or operation at hand. Simple window and icon designs, with unnecessary details results to less amount of time needed to complete tasks (Benbasat & Todd, 1993).

The second factor to consider is the proper groupings of display of information. These grouping will lead to the information's readability and the relationships among the information that can easily be recognized (Helander, 1988). One way of grouping is to classify tasks within icons. Icons are usually small and do not consume much screen space (Sears, 1993). Icons are also fast, easily recognizable and more visual than text (Benbasat & Todd 1993; Mandel, 1997). Novice user thus will find it easier to learn a system.

The third factor is the sequencing of information on display. The layout of information should be in a manner that it is easy for users to navigate to find the information he/she needs to perform the tasks at hand. For example, users mostly expect the top of screen will always contain the headings for the pull-down menus, and scrollbars are in either side of the screen. GUI designers should also consider more important information to be placed at prominent locations. The goal of GUI is thus to allow users just to use the application on the computer and to concentrate on primary cognitive tasks, not to be concerned with the user interface. If attention is devoted to the interface, this will interfere with the user's main tasks (Benbasat & Todd 1993; Mandel 1997; Galitz 1997).

2.4 3D Game Engine as a Tool for developing Virtual Reality Environments

All computer games have some form of engines working in the background. The engines can support 2D, 3D rendition of pixels (picture elements) or both 2D and 3D on the screen. There may be more than two engines working together and these engines are the backbone of the games. Computer games are computer programs consists of instructions (or lines of codes) that can accept input from a controller (usually from a user using a joystick, mouse or keyboard). Based on the user's action, the controller feeds it back to the computer and then the computer displays the action onto the screen (either monitor or television). This human-computer-interaction (HCI) occurs fast, in real-time and is repeated many, many times in a second. Such programs that provide real-time input and output between the human action and the computer reaction are classified as being interactive. An interactive and immersive game environment is when one is playing, one is immersed into the games environment and is subjected to the rules provided and 'how to control' aspects and unpredictable outcome of the computer game (Co, 2006).

Computer games that include 3D environment use 3D Game Engines to produce and display images in real-time on the display device. These game engines provide most of the significant features of a game environment that can include 3D scene rendering, networking, graphics and scripting, etc. (Finney, 2007). Game engines allow for sophisticated rendering of a VE. Each game uses a different system to organize how the visual aspects of the game will be modeled and presented. This visual aspects have becomes increasingly important as games are becoming more focused on 3D VE, richness in textures and forms, and an overall realistic feeling, playability and user involvement with the game.

The continuing popularity of computer and video games has provoked the development and introduction of brand names 3D Game Engines. The cost of licensing of these 3D Game Engines can range from free to involving a huge sum of money. Some examples of free Open Source 3D Game Engines are Crystal Space, Ogre, Irrlicht and Panda 3D. Some 3D Game Engines are also affordable and meant for independent (indie) one-man developer or small companies. Examples include the C4Engine, the Torque Game Engine, DX Studio and Unity 3D. There are 3D Game Engines that can cost a fortune to license such as the Unreal Engine, Half-Life and Crysis.

2.4.1 Strengths of 3D Game Engine

The strengths of 3D game engines lie in the following criteria: real-time rendering, real-time walkthrough, interactivity, multi-participants, lighting and collision detection. This section discusses the strength and advantages of using 3D game engines.

2.4.1.1 Real-Time Rendering

Adding visual characteristics, such as shading, shadows, and textures make 3D models more visually realistic. The entire process of calculating the appearance of the 3D model - converting it to an entity that can be drawn on a two-dimensional (2D) screen and then displaying the resulting image is called *rendering* (Finney, 2007). Specifically, rendering is the process of converting the 3D mathematical model of an object into an on-screen 2D image. There are also various techniques to apply color to (or rendering on) these faces such as Flat Shading, Lambert Shading, Gouraud Shading, Phong Shading, Fake Phong Shading, Texture Mapping, Shaders, Bump Mapping, Environment Mapping, and Mip-mapping (Finney, 2007).

Real-time rendering also measures the performance of a 3D game engine. Mullen (1998) suggested benchmarking the performance of a 3D game engine running on a computer can be done by measuring the number of images generated on-screen per second (i.e. frames per second or fps). Normally, when a 3D model is developed, the complexity increases as the level of realism increases. This will increase the real-time rendering therefore dropping the frame-rate of a real-time walkthrough noticeably. When the frame per second drops from 15 to between 3-4 fps while displaying a 3D model, the situation is unacceptable for presentational purposes and makes inspecting the model disorienting and difficult (Miliano, 1999). Many modern 3D game engines can maintain an interactive 30 frames per second image rendering. They can also handle a large number of polygons in a single scene including lighting and texturing.

2.4.1.2 Real-Time Walkthrough

A walkthrough application allows users to feel as though “they are there”, walking through space, able to move upstairs, peering out windows, etc which also give them sense of scale. Realism and details in VE are achieved through the process of adding 3D qualities such as shadows, colors and shade variances. According to Campbell and Wells (1994), such are the criteria that make VE closer to reality because of the ability to allow "immediate, direct, and more intuitive control over a three-dimensional design". The ability of the game engine to allow for the development of real-time walkthrough applications can benefit the AEC industry. An owner of a construction project can freely inspect a virtual facility earlier. The owner can better set realistic expectations on the final product, rather than just viewing representations in the forms of 2D drawings, static image rendering or fixed-path animation (Shiratuddin & Thabet, 2003).

2.4.1.3 Interactivity

The term interactivity refers to interaction between computer and user which takes place through changes of location views, typed commands, voice commands, mouse clicks, or other means of interfacing. The game engine accepts and responds to user activity in real-time at an interactive rate of 30 frames-per-second. This feature is incorporated and can be visualized and experienced in the VE of the game environment. Interactive features are important for users of the VE as it will relate what they are seeing in the VE to the real world. This feeling of realism is important to convince users the environment is realistic and represent the real world (Mays, 1998 and Miliano, 1999).

Interactive controls are used either by clicking on them or click-dragging the mouse across them. Some controls, like edit boxes, also require user to type in some text from the keyboard. Some of the controls have built-in labels that identify their purpose, and some will require you to create an accompanying non-interactive control to provide a label. Non-interactive controls, as the name implies, are used to only display information and not to capture user input.

2.4.1.4 Multi-Participants

In a VE, avatars are user defined geometry forms (Vince, 1998) that can either have intelligent characteristics (i.e. AI characters or bots) or just simply present a virtual representation controlled by the user's input. Avatars can represent a user having a tour in a virtual facility in group or individually. Avatars can also be configured to become workers simulating the environment of a real facility. Many other events involving avatars can be simulated using the built-in tools provided by the game engine.

Distributed VE systems allow the opportunity for multiple users to collaborate in real time. Such synchronous collaboration can occur via text chat, voice communication and through interaction with shared design components. In this way multiple users can develop designs in a real time virtual simulation. 3D CAD systems do not allow for such real time collaboration but there are possibilities for users to collaborate asynchronously. Typically this is done through text annotation and the marking up of designs via whiteboard-type functionality.

Besides synchronous or asynchronous communication the capability of the software to support a range of file types is an important attribute affecting collaboration. Being able to import and export multiple file types is important given that consultants will often be using different software. In this respect 3D CAD has greater support than the typical VE systems. However, this scenario is rapidly changing now through the introduction of open-standard file format such as the file format following the IFC standard and the Collada (Collaborative Design Activity) developed by the Khronos Group (Kitchens & Shiratuddin, 2007).

2.4.1.5 Lighting

Many game engines have 'dynamic lighting' feature that resembles real world lighting. Lighting provides the sense of security and confidence in occupying or maneuvering an enclosed space. Dynamic lighting can be seen in the games environment when a weapon is fired where the blast of a fire can cast light off the walls and surrounding objects. The second type of lighting includes shadow effects, where the engine will automatically show any shadow that should occur in reality when light is blocked by an object. The third type of lighting that many game engines support is colored lighting.

2.4.1.6 Collision Detection

Collision detection is described as "the process of detecting when two or more objects (in a simulated world) come into contact with one another" (Maurina, 2006). Collision detection improves interactivity in a VE. Many current commercial VE tools require collision detection to be manually defined by the user during the scene-building process. By default, most game engines enables collision detection for solid objects therefore disallowing users from walking through walls, doors etc. Many game engines are able to automatically detect when user collides with solid or non-solid objects, thus users will experience the 'bump' effect as in real life (Shiratuddin & Thabet, 2002).

2.5 Conclusion

Effective design review process allows for a more comprehensive, accurate, high quality design and construction documents. This in turn, results in construction cost that is within the intended budget (Lutz 1990; East *et al.* 1995; East 1998; Spillinger 2000; East *et al.* 2004). It can be concluded the general content of design review follows fundamental patterns.

- Information is an important resource that needs to be collected, compiled, stored, available, and usable when needed.
- All design review executions have set forth the quality targets and objectives that needed to be accomplished.
- Design reviews propose improvements where necessary to improve the subsequent processes and developments. This is where lessons learned are valuable.
- Defining subsequent design review actions.
- Design reviews confirm the readiness of building/facility for the next stage of development.

The traditional approach of design review has always aim to guide reviewers in performing the review on the design produced. Various tools are used in performing design review. They are interdisciplinary checklists, light table, online review system, and physical mock-ups (Staub-French & Fischer 2001; Shiratuddin & Thabet 2003a). These methods are mostly manual, inefficient, and do not use potential technologies such as centralized information databases, information visualization, and intelligent retrieval of information. Besides geometric, numeric and textual design information presented in many design drawings, the design review process requires access to other information available through various sources including construction contracts, design specifications, building codes and standards, safety manuals, design checklists, etc. These sources of information are scattered, either in paper-based or electronic (e.g. on-line) format and are not linked. The design review process becomes multitasked which forces design reviewers to work harder and quickly in cross-checking and reference information during the review. This may force reviewers to sacrifice the thoroughness of their reviews (East, 1998).

In the construction industry, there are various ways to perform design review. An avenue that can be explored is the use of VE. Referring to other industries experiences, conducting design reviews in VE enable designers to assess and validate alternative designs more readily and inexpensively than to build physical mock-ups. Design review has become a major VE productivity application (Brooks, 1999), and large research (Badler *et al.*, 2002), as well as many commercial packages, offer virtual approaches to these tasks. Previous work has shown there is a huge potential for improving design review in construction with incorporating VE (Shiratuddin and Thabet 2003a; Shiratuddin & Thabet 2003b; Shiratuddin & Thabet 2003c). VE if incorporated in design review will improve the design review process because relevant design review information, from the various sources, can be embedded in the VE, allowing an integrated and effective design review process. The information is more visually presented, and would provide a common language for all design review team members.

CHAPTER 3

3 DATA GATHERING AND DATA ANALYSIS

3.1 Introduction

This chapter discusses the research style and research methods the author used in the gathering and analysis of the research data, and in achieving the objectives of the research. The author used Action research style (Fellows & Liu, 2003) that requires the author's participation such as conducting interviews to identify, promote and confirm the problems, promote the potential solutions and develop a prototype. The author synthesized three methods which include review of literature, perform a case study and dissemination of his progressive work. The case study is divided into two sections i.e. interviews with industry personnel and content analysis of design review documents. At the same time and while the research work was progressing, the author with the assistance from his advisor disseminate the author's progressive findings in conferences, conference proceedings and journal publications. Disseminating the author's work helped him to identify, promote and confirm the problems, and promote the potential solutions.

3.2 Research Style and Methods

The author referred to Fellows and Liu's (2003) description of five styles of research available to construction researchers; action research, ethnographic research, surveys, case studies, and experiments. They are summarized in Table 3-1. One or more data collection methods may be used within each of these categories; for example interviews and questionnaires may be used in both surveys and case studies. The word 'method' is used to describe both the overall research style, and their corresponding data collection methods (Fellows & Liu, 2003).

Style	**Description**
Action research	The researcher actively takes part in the process under study to identify, promote and evaluate problems, and promote potential solutions. Typically, this research is aimed at directly influencing the current practice. Prototype development is a form of this research (Wroe, 1986) as cited in (Whyte, 2000).
Ethnographic research	The researcher becomes part of the group under study. However researcher's only become a passive observer (Leedy & Ormrod, 2005). The researcher spends a significant amount of time in site-based fieldwork such as 'going native' and engaged in the natural group setting.
Surveys	The researcher conducts a survey to assess the normative behavior of a sample through questionnaires and interviews (Oppenheim, 1992).
Case studies	The researcher studies a particular instance within a research subject. A case study is as an empirical inquiry that investigates contemporary phenomena within its real-life context, when the boundaries between

	phenomena and context are not clearly obvious, and in which multiple sources of evidence are used (Yin, 1994).
Experiments	Researcher tests relationships between identified variables within a controlled environment such as a laboratory. The effects of a change in the independent variables on the dependent variables are assessed (Fellows & Liu, 2003).

Table 3-1 - Research Styles derived from Fellows & Liu (2003)

Figure 3-1 summarizes the approach undertaken by the author for this research. The author used *action research* as the style of research. Action research is one of the most common types of research described in construction research (Fellows & Liu, 2003). The author compares his method to Whyte (2000). Whyte synthesized different methods to compensate for some of the limitations of the individual methods. Following White, the author uses three different methods in establishing the requirements to be included into the development of the design review approach.

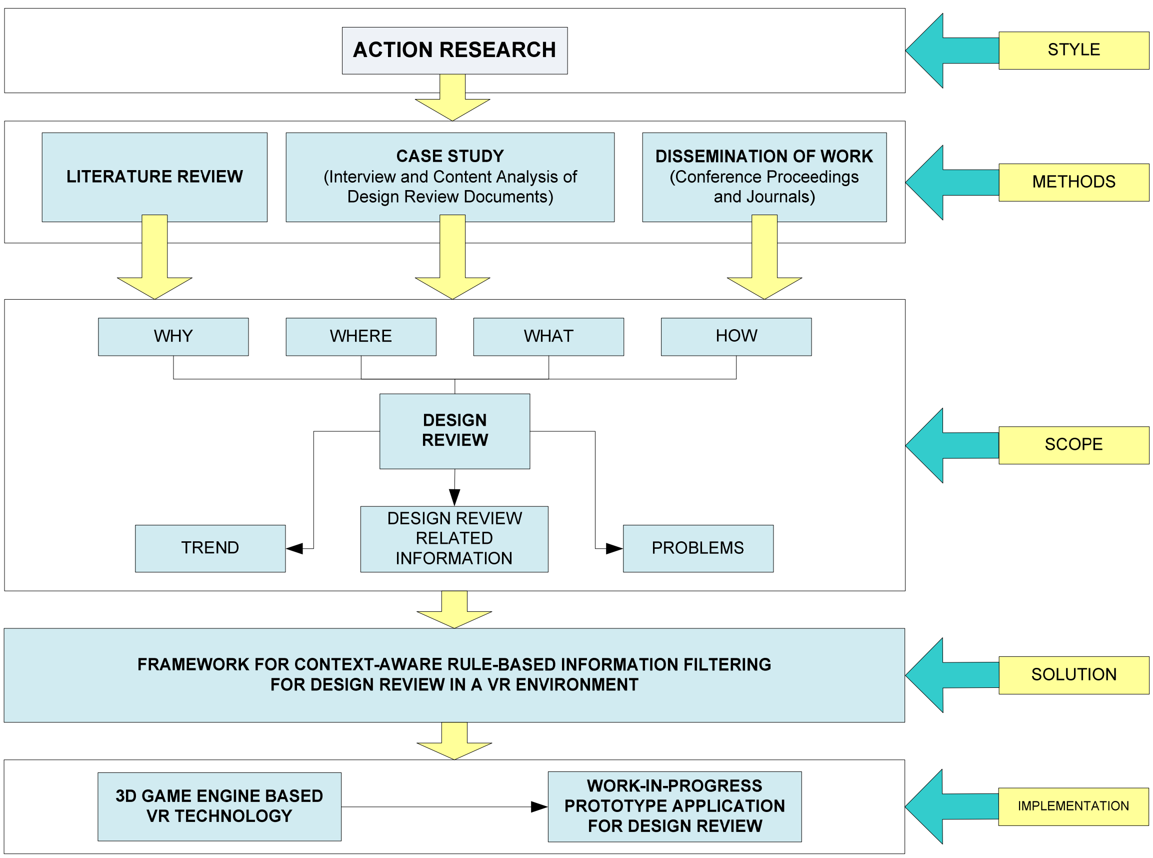


Figure 3-1 - Research style and methods undertaken by author for this research.

In collecting the data; literature review, a case study and publication of research papers in conference proceedings and journals were used to investigate different aspects of the research problem. In analyzing the data the author:

- Reviewed topics relate to: background of design review in construction, design review information, implementation of VE in construction, characteristics of VE, VE technologies, and VE specifically on information visualization.
- Conducted a case study. The steps taken for the case study were semi-structured interviews, and a content analysis of design review documents of three Virginia Tech's past projects. These projects are: a) The Career Services Facility Building, b) The Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech Complex, and c) The Bioinformatics Phase 1 Building.

The scope is the discovery of the "why, where, what, and how" of design review. It includes understanding of the trends of design review information, the prevailing problems of design review, the potential solution to the problems, and the potential of using VE technology to support design review. The author finally developed a work-in-progress (WIP) prototype design review application in a VE. VE technology was used as the driving tool in the new and improved design review application. A 3D Game Engine was used as the enabling development tool to develop the WIP prototype design review application in a VE.

3.3 Data Gathering and Data Analysis

3.3.1 Findings from Literature Review

The research began with review of literature that explored the current state of design review in the construction industry. According to Fellows and Liu (2003), it is essential that a search is carried out in the early stage of any research to identify potential relevant theory and literature. Following Leedy and Ormrod (2005), the author used literature review to:

1) Explore new ideas, perspectives, and approaches that may not have occurred to the author before
2) Inform the author about other researchers who conduct work in the same area
3) Show the author how other studies have handled similar methodological and design issues
4) Reveal sources of data the author may not have known existed
5) Introduce the author to measurements that other researchers have developed and used
6) Reveal methods of dealing with problems situation that maybe similar to difficulty the author was facing
7) Help the author interpret and make sense of his findings and tie his results to the work of those who have preceded him
8) Reinforce the author's confidence that his topic is one worth studying and there are others who have invested notable time, effort, and resources in studying it.

Review of literature reveals that design review has a main objective of making sure the project's priorities are met. Design reviews have been conducted by engineers and designers in many industries since the history of their business can tell, the only difference is the ways the design reviews are carried out (Ichida, 1996). The industries will try to do everything possible to resolve problems as they arise, and not to pass these problems downstream (Ichida, 1996; Pugh, 1991; Syan & Menon, 1994; Blanchard *et al.*, 1995). In construction, studies on design review by Lutz (1990), East *et al.* (1995), East (1998), Spillinger (2000) and East *et al.* (2004), insist that effective design review process during the design stage leads to more comprehensive preparation of accurate and high quality design and construction documents. According to Mitchell (1994), design review also may lead to optimal design because it is the platform where information exchange, interaction, and conflict resolution can take place.

The current process of overlaying 2D drawings on a light table is time-consuming and inefficient (Staub-French & Fischer, 2001). Design conflicts went undetected as 2D drawings do not adequately represent the spatial information and requirements of the building components. The light table and the interdisciplinary checklist are purely manual, and relied heavily on paper-based formats. Manual cross-checking of drawings and documentations to seek required information is still common. The author reviewed some of the design review system in the market. They are:

1) the Redicheck system which uses an interdisciplinary coordination paper-based checklist and overlay checking process (redicheck-review.com, 2006)
2) the Reviewer's Assistant (East *et al.*, 1995)
3) DrChecks/Construction Lessons Learned (CLL) software program which is used to conduct project reviews by recording review comments and designer annotations (East *et al.*, 2004)
4) iReview which is an online review and comment service system (http://www.ireview.com)

The method of using web-based online review system, involves perusing web pages, which is to some extent comparable to paper-based format, only substituted electronically by the computer and have some information storage and retrieval capabilities. The method of using Physical Mock-Up (or PMU) is expensive and time-consuming to both the project and team members, and lack of efficient way of handling information centrally and efficiently. The current methods of design review also lack a central coordination and integrated information source that manages and ensures the fluent flow of information. Design review related information is still at large placed at diverse places.

One of the problems the author discovered was the "scattered" nature and inefficient organization/arrangement of design review information for its availability, exchange, storage, and retrieval. Information is spread out resulting to the difficulty for accessing particular information at the right time, at the right place, and to and from the right people or source. Despite these challenges, the irony to this is that design review has to be performed within limited time periods with constrained resources. The scattering of design review information is also the cause of design deficiencies (Lutz, 1990; Lam & Chang, 2002). The potential solutions for this problem are the use of an effective information and data management system (Emmitt &

Gorce, 2003; Lam & Chang, 2002; Baldwin *et al.*, 1999), and a centralized information databases, information visualization, and intelligent retrieval of information (Shiratuddin & Thabet, 2007). Shiratuddin and Thabet (2007) describe four methods of filtering information for displaying relevant information to design reviewers in a VE. The benefits of this work are:

1) the generation of design review information relevant to a specific reviewer or specific tasks
2) help direct/navigate design review activities
3) display possible solutions to the reviewer based on lesson-learned gathered from past projects
4) inform reviewer if changes made will affect other adjacent or related components

Designers are responsible to ensure owner's requirement and description of his desired facility are translated into a representation that can be shared by everyone involved in a project. The problem with the current trend is this representation is created in 2D drawings either paper-based or electronically. Designer produce these drawings based on his 3D mental model visualized in his mind (Dunston *et al.*, 2003; de la Garza & Oralkan, 1995). The design reviewers' have to interpret these 2D drawings before design can be reviewed. Other project teams must also spend time mentally relating the project information to support decision-making tasks (Liston *et al.*, 2000). Even though the 2D drawings are packed with information (in the form of geometric, numerical and textual information), the process of interpreting 2D drawings is not perfect. 2D drawings do not adequately represent the multi-spatial information of a building in a more intuitive way, as 3D can. Only the designer can truly visualize what his creation look like in 3D and in his mind. Thus, there is a need to represent designs not just in 2D but 3D, where designers, reviewers, and other project team members can view the same model of a facility. Designers can also be sure the designs they produce is comparable to what the owner envisions.

3D is better than 2D in terms of the visual medium (Van Dam *et al.*, 2000), however, there are still some restrictions for both. A 3D real-time VE, however, has the ability to make the most effective use of data and information. It allows 2D information to be displayed with the 3D spatial model thus increasing understanding of the user (Liston *et al.*, 2000; Ganah *et al.*, 2001). Previous work by the author has shown there is a huge potential for improving design review in construction by incorporating 3D real-time visualization in a VE (Shiratuddin & Thabet, 2003a; Shiratuddin & Thabet, 2003b; Shiratuddin & Thabet, 2003c). The incorporation allows for important design review information from the various sources to be embedded in the VE. The information can be visually presented and provide a common language for all designers and design reviewers.

Galitz (1997) and Mandel (1997) emphasize the importance for well designed information display because it is the visible components that a user connects/relates to. Even with the outstanding features, a system will not be successful if it is not functional to the users. Following Mandel's suggestion on principles of user interface design, the author ensures that the new design review approach:

- Place users in control of the interface: In the design review mode, the main menu is designed to be visible at all-time. Meaningful commands are placed in the main menu e.g. File, View, Checklist, References etc. For the ease of navigating in the VE a keyboard and mouse combination is used. Using the W, S, A, D keys and the mouse, users are free to go where they want to go in the VE

- Reduce users' memory load: The computer can assist the user to remember and carry out tasks that are repetitive. E.g. defaults such as, undo, redo, copy, and the use of visible and meaningful icons
- Make the user interface consistent: Consistency is a key factor of a usable interface where users reuse their knowledge from other familiar software applications. The prototype application implemented by the author e.g. uses drop-down menus and right-click-popup menus. Many AutoCAD users are familiar with such concept.

Helander (1988) emphasized the importance of presenting information to the users. Following Helander's emphasis, the author ensures that:

- Information is displayed to users in less crowded manner so information clarity and readability is improved. The interface designed should only display the information the user needs to perform his/her task or operation. E.g. the use of simple window and icons with unnecessary details can result to less amount of time needed to complete tasks (Izak & Todd, 1993).
- Information is displayed in proper groupings so users can easily recognize the relationships of information. E.g. the use of icons that are small and do not consume much space (Sears, 1993).
- Information is displayed in logical sequencing. The layout of information is made in such a way that it is easy for users to navigate to find the information he/she needs to perform the tasks at hand. E.g. in a Microsoft Windows environment, users mostly expect the top of screen will always contain the headings for the pull-down menus, and scrollbars are in either side of the screen.

3.3.2 Case Study

The case study consists of two parts, interviews and a content analysis of design review documents of three Virginia Tech's past projects. These projects are: a) The Career Services Facility Building, b) The Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech Complex, and c) The Bioinformatics Phase 1 Building.

3.3.2.1 Interviews with Industry

The purpose of the interviews is to get an idea of the range of responses on thoughts or opinions that a group of relevant people have. This research is not interested in statistical analysis of responses in the form of what proportion of population gives a particular response. A *purposive sampling* method in selecting the design review related personnel was used. According to Leedy and Ormrod (2005) in *purposive sampling*, people or other units are chosen for a particular purpose. A researcher chooses people who are "typical" of a group or those who represent diverse perspective on an issue. E.g. to forecast elections, agencies choose a combinations of voting districts that, in past elections, have been useful in predicting the outcomes. The essential criteria for selection of a sample for the interviews was the sample is knowledgeable about design review issues and involve in design review in the AEC industry.

The author then had to decide the method of interview to be used. According to Fellows and Liu (2003), interviews can be constructed in three ways; structured, semi-structured, and unstructured. The major differences lie in the constraints placed on the respondent and the interviewer. Leedy and Ormrod (2005) explain that in a structured interview, the researcher only asks a standard set of questions. An unstructured interview is more flexible, and the researcher is more likely to yield information that is more than necessary (Fellows & Liu, 2003). In a semi-structured interview, the research may follow the standard questions with one or more individually tailored questions to get clarification or probe a person's reasoning (Leedy & Ormrod, 2005). The author selected a qualitative semi-structured interview method. No quantitative data has been gathered as this is not the goal of the research objectives.

10 interviews were conducted. The interviews were with local and national architecture and construction companies. Since the research uses case studies from three of Virginia Tech's past projects, following interviews and discussion were conducted with the Virginia Tech's Office of the University's Architect and the Capital Design and Construction at Virginia Tech to gather more design review information about the projects. Interviews and phone conversations were also conducted with a local architect in Hattiesburg, Mississippi. The local architect in Hattiesburg has agreed to assist the author in testing and giving out feedback during the development of the prototype design review application.

Table 3-2 shows the breakdown of the companies involved in the interviews. To protect the identity of the companies, actual names of the companies are not used in this dissertation. The names are replaced with generic labels. A semi-structured set of questions was used for the interviews.

Name	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**
Type	Arch	Arch	Arch & Eng	Arch (Univ)	Home Builder	Owner Rep (Univ-PM)	Home Builder	Arch (Navy)	Arch	Arch
# of personnel interviewed	1	1	1	2	3	2	4	1	1	1
Project Type Supported	DBB	DBB	DB	DBB	DBB	DBB	DB	DBB	DBB	DBB

Table 3-2 - Breakdown of the companies interviewed by the author

The questions are as follows:

1) What are the design stages in your company?

This question is to inquire the design stages in the company. For Design-Bid-Build (DBB) projects, all companies follow the traditional stages of design i.e. Programming, Preliminary Design, Schematic Design, Design Development, Working Drawings and Construction Document. However, three of the companies also do work using Design-Build (DB) and therefore a more concurrent design approach is used. In a DB delivery system, partially completed designs are handed out to the contractor to allow for construction. These designs are usually sufficient for the contractor to begin work.

2) What are the tools used during design and design review?

This question is to inquire the tools used during design and design review. Some companies use pencil and paper during Programming and Schematic design stage. 2D CAD software was used during Design Development where designs were more refined with more definite scope. During design review, printed copies of the design were used. Markups and comments were made using pencils, pen and/or highlighters.

The results are shown in Figure 3-2. During Programming and Schematic Design stages, seven companies use pencil and paper, and three companies use 2D CAD. During the Design Development stage, all companies use 2D CAD. During design review all companies use printed copies of the designs, and make markups on them.

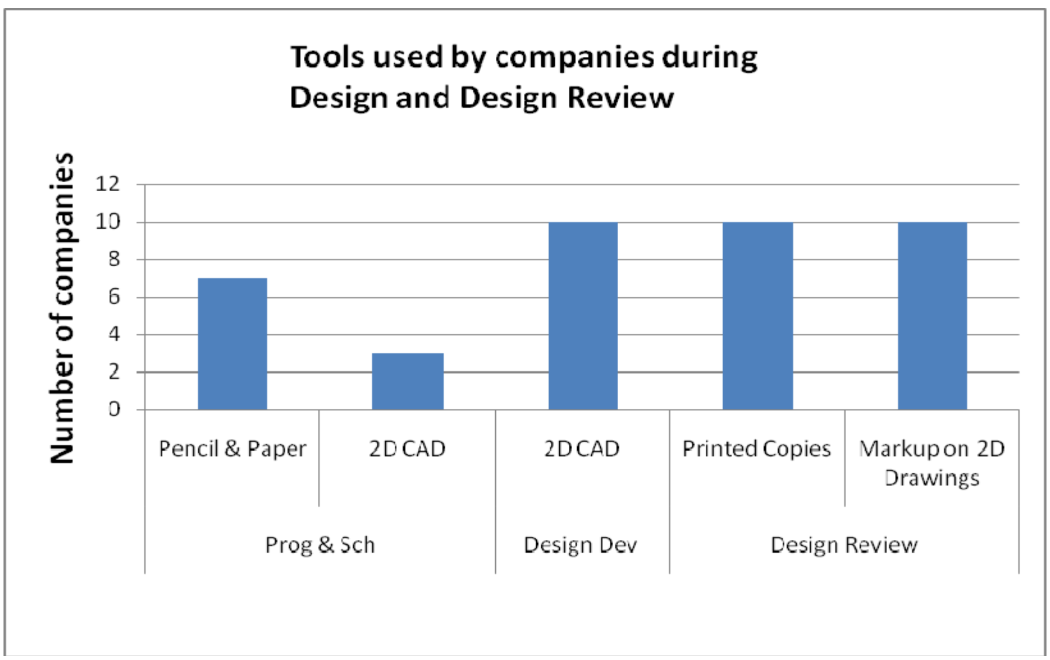


Figure 3-2 - Tools used during design and design review

3) How do you communicate with your client?

Methods of communicating with owners can be face-to-face, letters, phone calls, faxes and web conferencing, and any combination of the above. The results are shown in Figure 3-3. All companies use face-to-face, letter, phone call and fax as method of communication. Only three companies use web-conferencing.

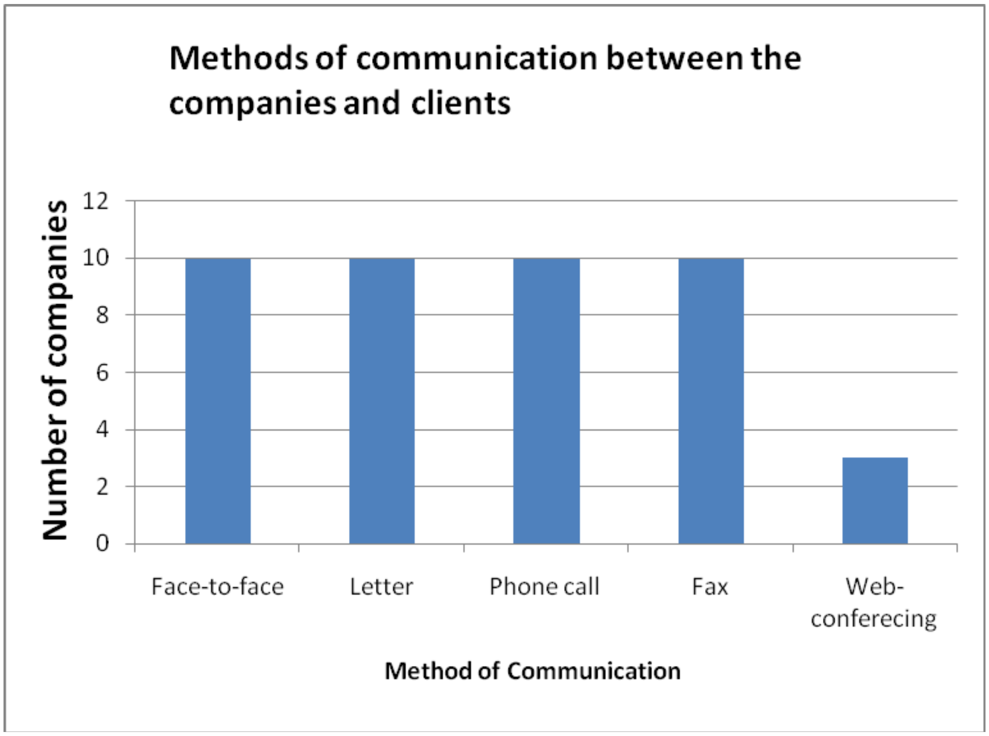


Figure 3-3 - Method of communication

4) Do you involve contractor in early design stage?

In DBB projects, all design companies do not involve contractor in early design stage. A comment made by one of the architect during design review, they will try to think and play the role of a contractor. They will ask whether there is enough information on the drawing sheet to allow them to build what the owner wants. However, one company said that if the owner hired the designer and contractor at the same time, then the general contractor and major subcontractor is 100% involved from day 1. Two builders (contractor-led) do include contractors and subcontractors during the design stage.

5) How do you perform design review?

This question was asked to seek how the company performs design review. From the interviews, all did their design review in-house using 2D hard copy drawings with specifications and design analysis. The design analysis includes written narrative of e.g. finishes, materials, structural system, MEP etc. Majority did manual reviews by comparing one design sheet with another. One company used light table for their design review. Figure 3-4 shows none of the companies used 3D CAD model for design review.

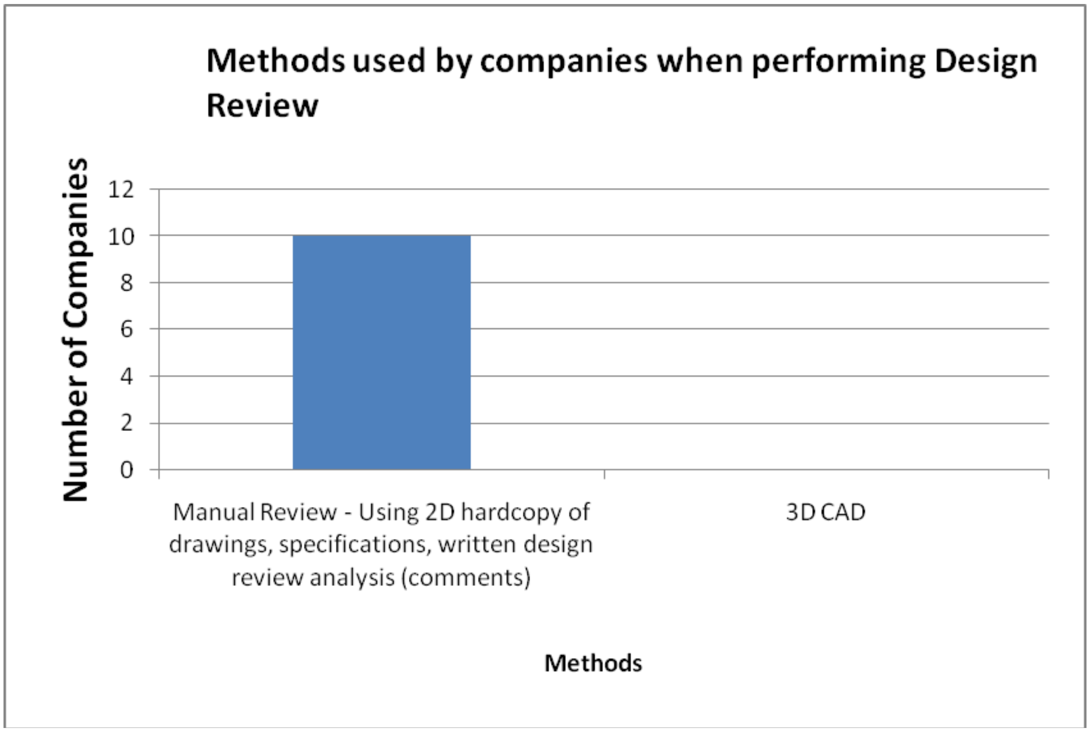


Figure 3-4 - Methods used when performing design review

6) How do you resolve conflicts if found during design review?

In all companies, if conflicts were found during review, identified markups on the 2D CAD drawings were made and comments were documented. Conflicts were resolved through further discussion with the designer in-charge.

7) What do you provide the owner during design review?

All companies provide the owner with 2D paper drawings and specifications (Figure 3-5).

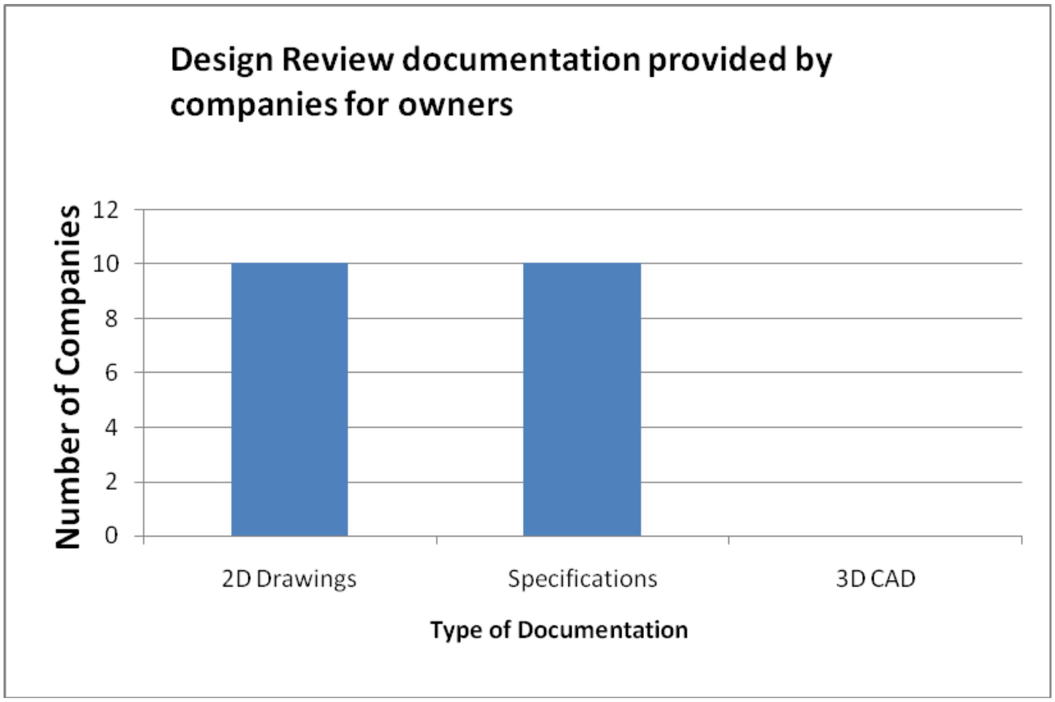


Figure 3-5 - Types of design review documentation provided to the owner

8) Does the owner fully understand the 2D drawings?

Most owners understand the 2D CAD drawings up to a certain degree. If there are identified misunderstandings, they are cleared up immediately through the design review to avoid confusion and provide possible cost implications.

9) Do you have a checklist for design review?

The question is to seek whether a checklist was used during design review. All the companies have their own in-house design review (with some calling it quality assurance checklist) checklist. A copy of the checklist was obtained, and from the checklist it can be clearly seen there is not a standard checklist for design review.

10) Do you use any 3Dmodels?

Only one company uses 3D model. Again it is still limited to the types of project they are involved. Sometimes the 3D models are required as part of the deliverables either for the contractor or owner.

11) Do you have any suggestions in improving your current design and design review approach?

In all the companies interviewed, their current design review approach mainly uses 2D CAD hard copy drawings. The drawings were cross-compared with one another and references were referred to as needed. Design review was performed in-house. However, if there are other parties involved e.g. the owner and specialty designers, then a face-to-face meeting was conducted. Communications through telephone and fax was also a means to conduct short design review session or confirmation of information.

Three main suggestions were extracted from the interview on improving design review:

a) Use 3D model from the beginning. The 3D model must also be able to produce 2D plans for construction documents.
b) Better ways to communicate, collaborate, share and view the design in real-time. One company suggested videoconferencing while another used web-conferencing service (through Microsoft Netmeeting).
c) Better methods of accessing relevant design review information. All the companies agreed that design review information is too scattered all across volumes of manuals. Some used online databases to access and retrieve design review related information, but much of the information is still in printed copy format that needs to be referred to with no or minimal search features. Some have their own internal databases but access to other information outside the system is not possible.

The conducted interviews has assisted the author to better understand and confirmed the real-challenges facing design review. The interviews identified the general trends of design review, subjects' view on design review in a VE, and the subjects' wish-list, suggestions and recommendations on areas of design review that needs improvement etc.

An advantage the author discovered during the interview is the general "feel" of the interview as suggested by Leedy and Ormrod (2005), because of a face-to-face interview between the author and the interviewees. The interview sessions felt informal and friendly. In contrast, Leedy and Ormrod, and Fellows and Liu state that if the interviews were a quantitative study, the sessions would be emotionally neutral. During the interview, the interviewees felt that they were engaging in a friendly chat with the author, who was someone they had come to know and trust. The author also managed to gain interviewees cooperation and encourage them to respond honestly. This was because the latter knew the author was genuinely interested in what the interviewees have to say. Leedy and Ormrod also state that this interview usually yields the highest response rate in the percentages of people agreeing to participate. However, as the author experienced, the restricted time and finance of this research have prohibited more number of interviews to take place.

3.3.3 Content Analysis of Design Review Documents

In gathering the information that is required for design review, the author sought cooperation with the Virginia Tech's Office of University Architect (OUA). The OUA was able to provide limited assistance on a *pro-bono* basis. The author was furnished with complete sets of design review documents for three of Virginia Tech's construction projects. These projects are 1) The Career Services Facility Building, 2) Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech Complex, and 3) The Bioinformatics Phase 1 Building.

The author used “Content Analysis” method to extract design review information from the design review documents provided by the OUA. Content analysis is a systematic, replicable technique for compressing many words of text into fewer content categories based on explicit rules of coding (Krippendorff 1980; Weber 1990). Krippendorff (1980) outlines the history of content analysis in his book. He traces its development from its origins in Renaissance analysis of religious texts, through early 20th century focus on newspaper content, World War II concerns with propaganda, and postwar expansion into broadcast media and advertising. According to The Division of Instructional Innovation and Assessment (2006), existing documents and data are excellent sources of information because they are records of events and practices and do not rely on participant memory or predictions of future behavior. The systematic examination of existing documents is useful for gaining insight into an activity or approach, examining trends, patterns, and consistency in documents.

3.3.3.1 Selected Virginia Tech’s Construction Projects

Three projects were selected by the author for this research. The selection was based on the accessibility of information that can be provided by Virginia Tech’s building authorities. The three projects were: (1) Career Services Facility Building, (2) The Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech and, (3) Bioinformatics Phase 1 Building.

a) Career Services Facility Building

Figure 3-6 - Career Services Facility Building

The Career Services Facility (CSF) (Figure 3-6) was built to provide the space needed to accommodate the needs of students and employment recruiters. The facility is also intended to improve the university’s ability to meet the employment requirements of the students and prospective employers. The facility is a two-story building with 21,735 sq.ft, and the duration of the project was Winter 2003 to Spring 2004. The project budget was $4,608,000. The Architect/Engineer for this project was Hanbury Evans Newill Vlattas of Norfolk, Virginia. The General Contractor was Avis Construction Co., Inc. of Roanoke, Virginia.

b) The Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech

Figure 3-7 - The ACECHC during construction

Figure 3-8 - The ACECHC

In this case study, the Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech (Figure 3-7 & Figure 3-8) is called the ACECHC. ACECHC is the abbreviation used in the design review documents when the complex were not yet given specific names, and was called the Alumni Hall, Continuing Education Center & Hotel Complex (ACECHC). It is four-story high, with total area for the complex is 193,000 sq. ft. and the total site area is 25 acres. The duration of the project was from April 2003 to September 2005. The project budget was $43,118,000. The Architect/Engineer for this project was Sheretz Franklin Crawford Shaffner, Inc. of Roanoke, Virginia. The General Contractor was Branch & Associates, Inc. of Roanoke, Virginia.

The Alumni Hall was later named the Holtzman Alumni Center. It is in the east wing of the complex. It serves the campus “home-away-from-home” for the university's 190,000 living alumni. The Holtzman Alumni Center shares the Assembly Hall with the adjacent Skelton Conference Center and has a direct connection on the second level. Its paved donor terrace adds more outdoor space for alumni events. The Skelton Conference Center is the central wing of the

Alumni Center and the new hotel. It has 23,705 square feet of appointed meeting space equipped with state-of-the-art technology, and is the largest of such facility in Virginia west of Roanoke. It has the capacity to host conferences of up to 1,250 people with the 700 banquet seat Latham Ballroom (divisible into six smaller spaces), 10 comfortable conference rooms, and the Alumni Lawn for outdoor functions. The inside function and outside terrace spaces, when used together, will accommodate several thousand people. The west wing of the complex is the hotel called The Inn at Virginia Tech.

c) Bioinformatics Phase 1 Building

Figure 3-9 - Bioinformatics Phase 1

The Bioinformatics Phase 1 (BP1) (Figure 3-9) is part 1 of 2 of the Virginia Tech's Virginia Bioinformatics Institute project. The Institute can provide more than 130,000 square feet of state-of-the art office, conference and laboratory space. The BP1 is three-story high with 60,000 gross sq. ft. area, which includes an enclosed mechanical penthouse above, and the project budget was $13,527,313, from Spring 2001 to Fall 2003. BP1 composed of general and computational labs, a core lab including sequencing, microarray and proteomics functions, computer servers for data analysis, and administrative and research office space. The Architect/Engineer for this project was Calloway Johnson Moore & West of Richmond, Virginia. The General Contractor was Branch & Associates, Inc. of Roanoke, Virginia.

Table 3-3 shows a summary of the scale of the projects.

Project	***Career Services Facility Building***	***Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech***	***Bioinformatics Phase 1 Building***
Budget	$4,608,000	$43,118,000	$13,527,313
Area (square feet)	21,735	193,000	60,000

Table 3-3 - A summary showing the scale of the projects

3.3.3.2 Applying Content Analysis to Construction Design Review Documents

The following steps were taken by the author in applying Content Analysis method in extracting design review information from the said documents. These steps are following Krippendorff (1980) and Weber (1990):

1) Each document is read once with brief notes are made in the margin whenever something containing relevant information comes up.
2) Margin notes are reread and a list of different types of information that have been found is made.
3) The list of data items is then categorized into like items. The author now has a list of items excerpted from the text. The list of data items is the read through and each item is categorized in a way that describes what it is about.
4) The categories are then considered again whether some categories may be linked in some way. These categories are then listed as major categories and minor categories.
5) Once all the categories are sorted out, and all the items of data are in the right category, the range of categories is compared, to see whether two or more categories fit together. If so the categories that fit together form a major category in the analysis.

The author was provided by the OUA with electronic copies of drawings in form of either AutoCAD *.dwg, AutoCAD *.dwf file format, and design review documents either in Word *.doc or *.PDF files for each project. These files can be further categorized into Schematic, Preliminary and Working Drawings. In practice, for each stage, following a standardized checklist, design reviewers review the 2D drawings and then typed-in their comments in a word document or form.

Figure 3-10 shows an example of a design review document for the Working Drawings phase of the Career Services Building. The general information that can be derived from this document are the name of project, date, the stage of the design review, the name and the discipline of the design reviewer, the comments made by the reviewer and a space for the A/E to respond to the comments. For these activities, the Project Manager acts as the intermediary between the A/E and the reviewer, and passing the documents through the right channels, to and from the right personnel/department. Comments were provided by the reviewer from the Civil/Architectural/Structural discipline.

VIRGINIA TECH-FACILITIES DEPARTMENT
INTERNAL REVIEW **Page 1 of 4**

SUBMITTAL:	Working Drawings	REVIEW DATE:	June 12, 2002
REVIEWER:	John Beach, JGK	PROJECT CODE:	208-16477
DISCIPLINE:	Civil/Architectural/Structural	RECEIVED DATE:	May 23, 2002

AGENCY: VIRGINIA TECH
PROJECT: CAREER SERVICES BUILDING

#REF	REVIEWER COMMENTS	A/E RESPONSE

1. **COMMENTS—ROOFING:** In the Schematic Phase, we had expressed a preference for a Fully-Adhered Single Ply System in lieu of a Ballasted System. No specific references to a particular single-ply system were found in the preliminary stage. We would much prefer the fully-adhered system rather than ballast.

2. The A/E needs to check the specifications in all roofing sections against Section 707B of the CPSM. The following are offered as a courtesy,
but this may not constitute an all-inclusive list:
   a. Provisions pertaining to roof inspector should comply with Section 707B.5
   b. Section 707B.6.2 specifies that the Pre-Roofing Conference should take place before materials are ordered.
   c. I could not find reference to a tear-resistance specification as required by the CPSM. It is possible that the ASTM Standard cited as a general guide may cover this area.
   d. Comply with CPSM Section 707B.14.6 regarding roof protection.
   e. Comply with CPSM Section 707B.14.7 regarding final inspection procedures.

3. The use of a cold applied adhesive to attach "subsequent" layers of insulation as described in Part 3.3 (G) of Section 7531 is something that is being done in State applications as of present, but as the CPSM is currently written, this will require a waiver from BCOM.

Figure 3-10 - Sample of a design review documents from the Career Services project

Following reading/reviewing and re-reading/re-reviewing of the design review documents, sub-categories emerged and were then categorized and labeled as categories. Finally, this process revealed four main categories of design review information with each category has sub-categories and sub-sub-categories (Figure 3-11).

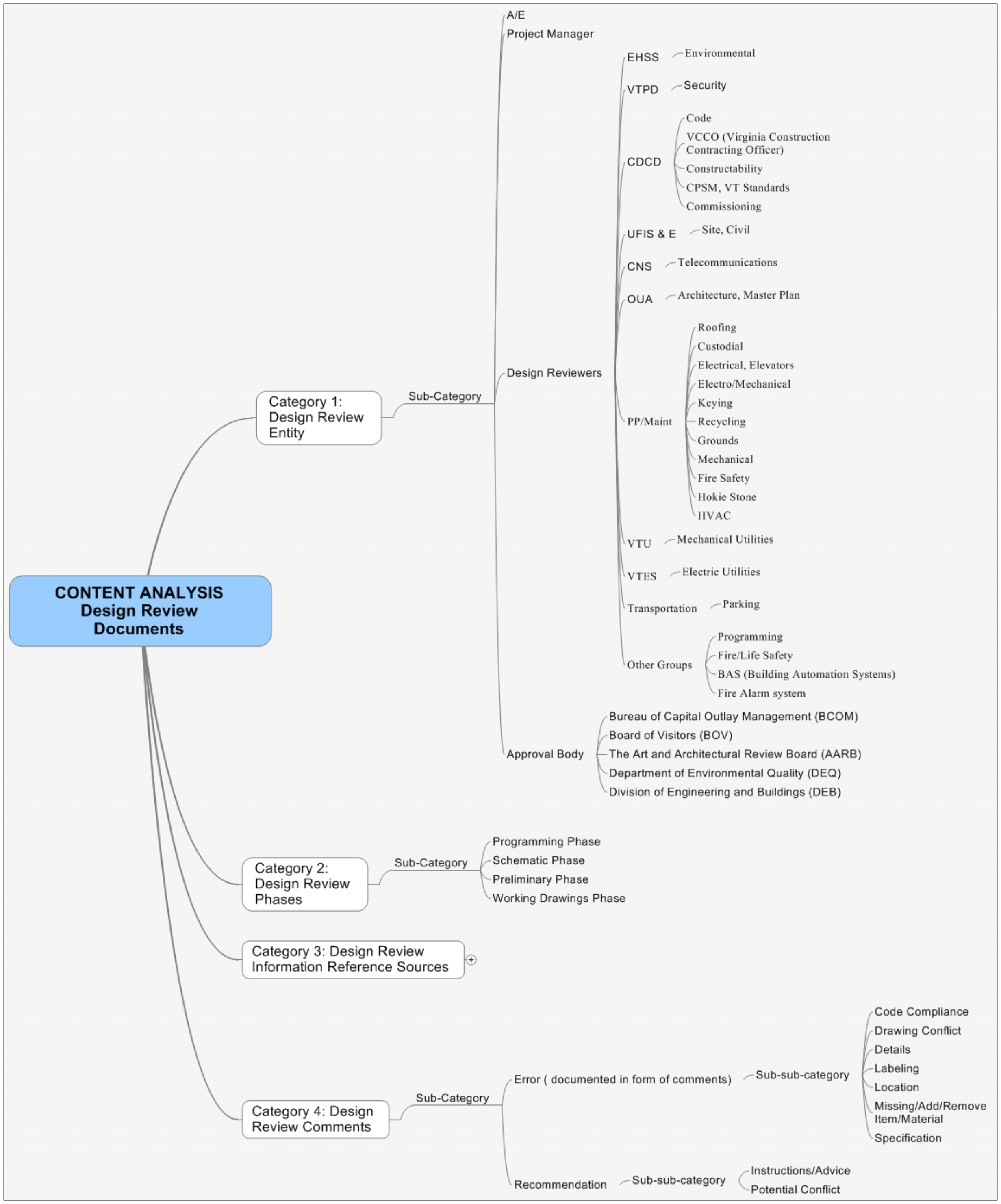


Figure 3-11 - Emerging categories, sub-categories and sub-sub-categories of design review information from the Content Analysis

Four main categories of design review information that emerged from the analysis are:

Category 1: Design Review Entity

The entities (sub-categories) involved in the design review process are: the A/E, the Project Manager, the Design Reviewers and the Approval Bodies.

a) The Architect/Engineer (A/E)

The A/E team is comprised of the Principal Architect, the Landscape Architect, the Structural Engineer, the MEP Engineer and the Civil Engineer. Based on their individual expertise, each individual member has the responsibility to prepare the design drawings for the Basis of Design set out by Virginia Tech. Basis of Design is the documentation of Virginia Tech's interpretation of the project requirements and where decisions that set the tone for the project are made. It is formally developed with the A/E on the A/E's selection as the principle designer for the project. Basis of Design is also the point of reference used by other A/E members. Important information is specified in the Basis of Design indicating the purpose of the project, any special circumstances, design considerations and goals of the project.

Some of the responsibilities of the A/E during design review are to:

- Use the right channel of communication to relay information with other essential parties in design review.
- Prepare and submit written responses to all reviewers' comments and, if applicable, provide the related technical data.
- Resolve all the outstanding issues, comments from reviewers, and the Value Engineering recommendations before proceeding with the Working Drawing Phase.
- Prepare the Final Drawings based on the accepted Preliminary Design documents, and resolved review comments to be included in the Contract Documents. Drawings must show the complete scope, extent and character of the work to be furnished and performed by the Contractor(s). Also, the A/E has to specify all finishes and provide color selections of all materials and finishes in the construction contract.

b) The Project Manager

For all Virginia Tech's construction projects, Project Manager (PM) comes from the Capital Design & Construction Department (CDCD) of Virginia Tech. The PM is also the point of contact for the project design review team to communicate information among the disciplines, Virginia Tech representatives and the Commonwealth of Virginia regulating agencies. CDCD is a department in Virginia Tech which is responsible for managing the process by which capital outlay projects are designed and constructed to effectively balance each project's established parameters for cost, schedule, and quality. The PM has the task of ensuring the project is well-coordinated and within the standards of Virginia Tech's capital projects.

The roles of the PM are:

- To resolve any potential conflict in interests that may occur among Virginia Tech, the environment, code and safety authorities, and many groups with special interests in the construction of the facility.
- To ensure the project complies with:
    - the Virginia Tech's Design Guidelines and Construction Standards, the Virginia Uniform Statewide Building Code (USBC),
    - the Construction and Professional Services Manual (CPSM),
    - the Higher Education Capital Outlay (HECO) Manual,
    - local, state and federal codes and state procurement and contract laws.
- As the key person responsible throughout the duration of the design and construction process and will be the principal contact for the current status and up-to-date information to Virginia Tech.
- To provide enough time for design reviews at the proper phases in developing the design documents.
- To ensure communication and proper documentation is properly distributed among design review team members such as:
    - distributing drawings and specifications from the A/E to reviewers
    - reviewers' comments are communicated to the A/E
    - the A/E incorporate proper changes requested by the reviewers to the drawings or specifications where necessary
    - to ensure the design documents are then distributed to the proper approval bodies
    - to sign off the documents to assume the next phase of design when the design for a particular phase is complete
    - Coordinate Design Review meetings

c) The Design Reviewers

The reviewers' task is to review and ensure the design documents are free from error. The reviewers ensure designs are prepared accordingly and confirm the project will be constructed based on the requirements of the Basis of Design, of materials, systems and characteristics. The reviewers also make certain the designs comply with current building and construction standards, operating procedures and maintenance requirements.

The reviewers were appointed by Virginia Tech. The roles of the reviewers are to:

- review that all standards are complied with in design, and in drawings and specifications
- ensure designs, drawings and specifications are free from errors, inconsistencies and ambiguity
- provide comments to any errors, inconsistencies, missing information to the A/E to be considered and when necessary incorporated in the design
- ensure designs, drawings and specifications will meet Virginia Tech's scope, fit budget and is constructible within the given time frame
- ensure designs, drawings and specifications will be accurate and complete for bidding
- ensure correct specification of the required materials for construction

Table 3-4 shows the areas of review the design reviewers will look at, the affiliation where the reviewers come from, and the disciplines where the reviewers represent and be responsible of.

	Department	***Discipline***
1	EHSS - Environmental, Health and Safety Services	Environmental
2	VTPD - Virginia Tech Police Dept	Security
3	CDCD - Capital Design & Construction Dept.	Code, VCCO (Virginia Construction Contracting Officer), Constructability, CPSM, VT Standards, Commissioning
4	UFIS & E (University Facilities Information Systems and Engineering)	Site, Civil
5	CNS – Communications Network Services	Telecommunications
6	OUA- Office of the University Architect	Architecture, Master Plan
7	PP/Maint - Physical Plant/Maintenance	Roofing, Custodial, Electrical, Elevators, Electro/Mechanical, Keying, Recycling, Grounds, Mechanical, Fire Safety, Hokie Stone, HVAC
8	VTU – Virginia Tech Utilities	Mechanical Utilities
9	VTES - Virginia Tech Electric Service	Electric Utilities
10	Transportation	Parking
11	Other Groups	Programming, Fire/Life Safety, BAS (Building Automation Systems), Fire Alarm system

Table 3-4 - The design reviewers

d) The Approval Bodies

The Approval Bodies taking part in the design review of all Virginia Tech's construction projects are as follows:

<u>Bureau of Capital Outlay Management (BCOM)</u>
BCOM is part of the Virginia Department of General Services, Division of Engineering and Buildings. BCOM publishes the Construction and Professional Services Manual (CPSM). CPSM sets the standards, policies, terms, conditions, and procedures for all institutions in procuring the design and construction of all structures on state property including renovations, modifications, and additions to existing facilities. BCOM reviews initial budget submissions to determine if the institution can construct the project as requested. BCOM also monitors projects throughout their life by reviewing and approving institutions' submissions as required by the CPSM. The specific intent of BCOM reviews during the design phase is to ensure the project meets the building code and reduce unforeseen changes later in the process.

<u>Board of Visitors (BOV)</u>
The BOV reviews the designs because it is in charge of:

- the care and preservation and improvement of the property of Virginia Tech
- the protection and safety of students and other persons residing on Virginia Tech's property

<u>Art and Architectural Review Board (AARB)</u>
The AARB encourages the design of buildings and works of art that are both aesthetically and functionally appropriate for Virginia Tech. Generally, AARB requires each submission of design to demonstrate:

- A resolution of basic functional and organizational requirements
- A command of the fundamental principles of good design, including refinement of color, form, scale, material, and craft
- A positive contribution to the order and aesthetic of the physical setting
- Due consideration of its environmental, historical, and cultural factors
- Concerns for the greater public good

<u>Department of Environmental Quality (DEQ)</u>
The DEQ reviews the Environmental Impact Report prepared by the A/E:

- The review assures that environmental impacts that require modification of a major state project are identified in a timely manner, before acquisition or before construction begins on Virginia Tech, which is an existing state land.
- DEQ determines whether any of their proprietary, management, policy, or regulatory responsibilities is likely to affect or be affected by the Career Services project.
- If permitting will be required, regulatory agencies may identify criteria or permit conditions to aid Virginia Tech in preparing a permit application.

Division of Engineering and Buildings (DEB)

DEB acting on behalf of Virginia Department of General Services (DGS) function as the code official and the building maintenance official for state-owned buildings like Virginia Tech. As the code official, the DEB has the authority for building code enforcement for buildings. As the building maintenance official, the DEB enforces the compliance with the repair and maintenance of existing structures and equipment. The purpose is to ensure public safety, health, and welfare through proper building maintenance, repair, and use and continued compliance such as accessibility and energy conservation.

Category 2: Design Review Phases

The design phase is a sequential process and can be broken down into 4 sub-phases (sub-categories) starting from Programming, Schematic, Preliminary & Working Drawings (Figure 3-12).

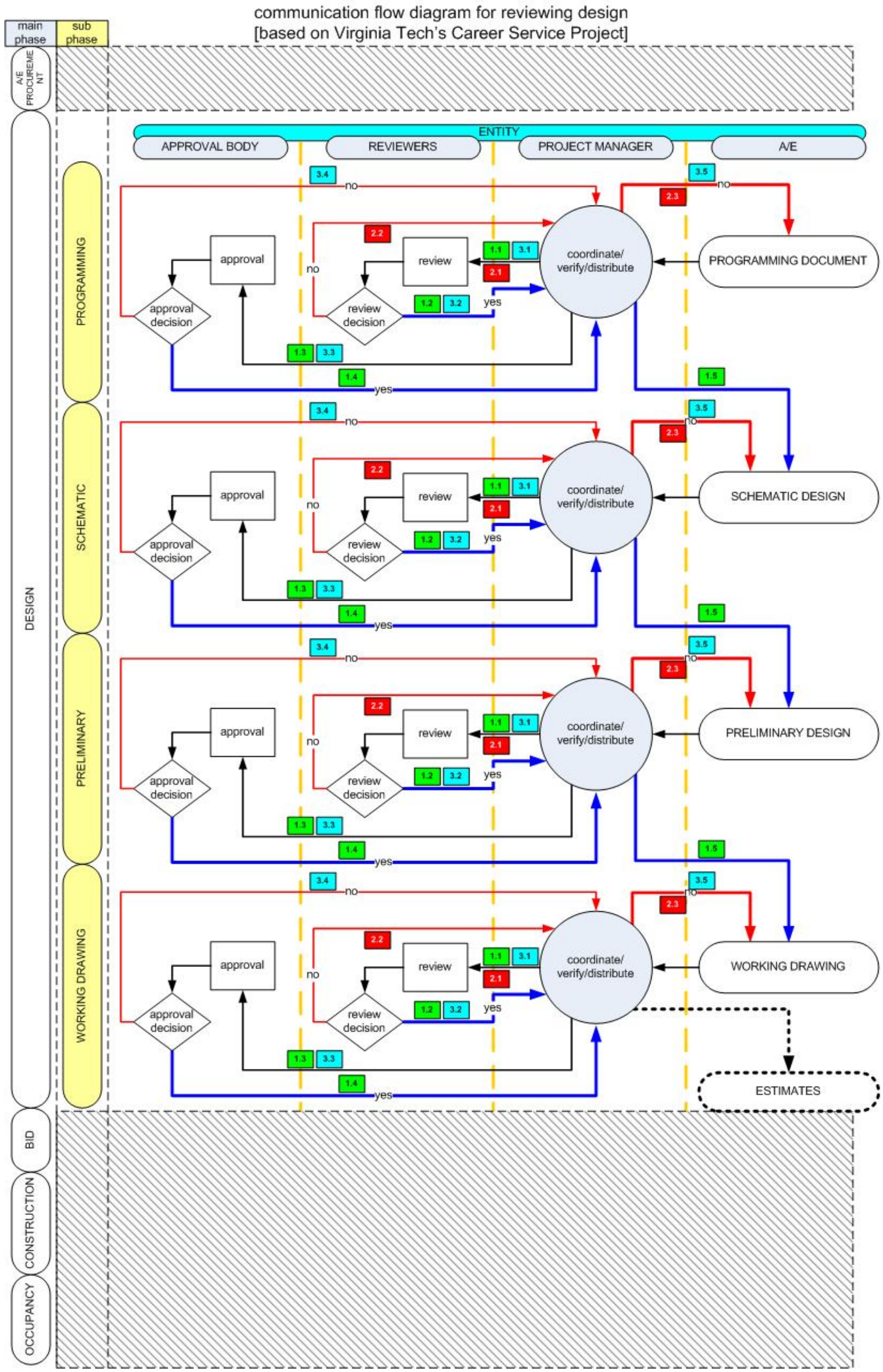


Figure 3-12 - The overall design review process

Referring to Figure 3-13 and taking the programming phase as an example:

- The A/E prepares the drawings required for the project.
- When the drawings are completed, the A/E passes the drawings to the PM.
- The PM then distributes the drawings to the respective reviewers (for the Career Services project, there were approximately 35 reviewers appointed from various disciplines;
- 4).
- Each reviewer is assigned to review certain discipline/s (e.g. mechanical system, electrical, architectural, communications etc.) and provide comments on any errors or inconsistencies found so the A/E can make the required modifications.
- Whether approve or disprove, the drawings are passed back to the PM. If there is/are error/errors identified and informed by the reviewers to the PM, the PM will give the drawing/drawings back to the A/E for amendments.
- If there is no error or inconsistencies, the PM then passes the drawings to the Approval Bodies for codes and regulations approvals.
- Whether approve or disprove, the drawings are then sent back to the PM. If the drawings are approved, the drawings are passed back to the A/E for the next design phase. However, if the drawings are disproved, the PM gives back the drawing/drawings to the A/E for further modifications.
- Once all amendments are made and the drawings are completed, the schematic design phase will commence.

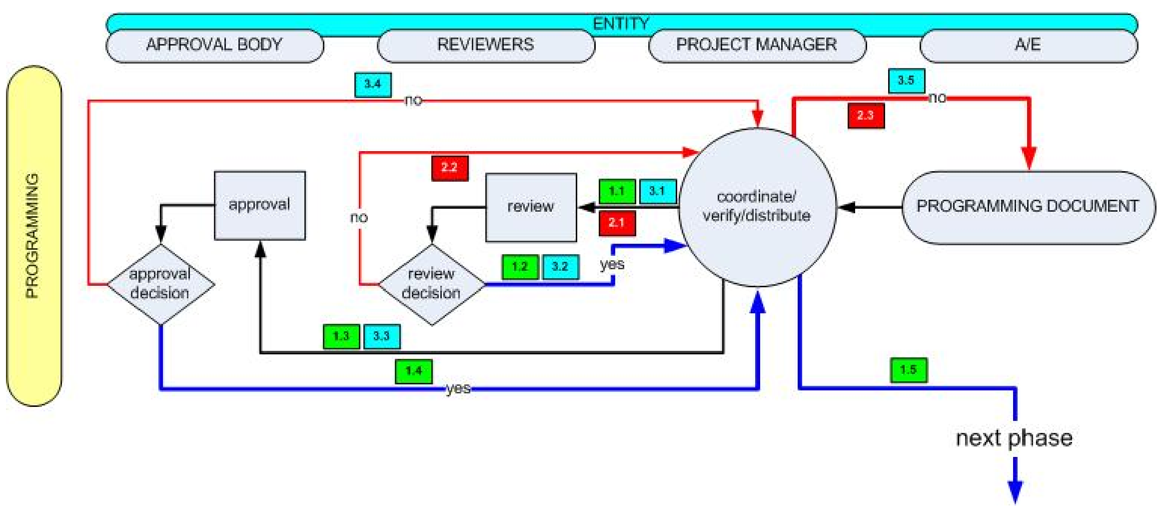


Figure 3-13 - The programming phase design review process

There are 3 possible routes (refer to the numbered color boxes) of the review process:

- 1.1 → 1.2 → 1.3 → 1.4 → 1.5 [design approve by reviewers and approval bodies]
- 2.1 → 2.2 → 2.3 [design disprove by reviewers; there are errors or inconsistencies; require redesign; design will not go to approval bodies]
- 3.1 → 3.2 → 3.3 → 3.4 → 3.5 [design approve by reviewers but disprove by approval bodies; require redesign]

The design review process for this project started with the Programming Phase and ended before the Bidding Phase began. In each phase, the PM and the design reviewers use design review checklists to carry out design review. These checklists are prepared based on the guidelines set forth by the BCOM. Figure 3-14 and Figure 3-15 below are examples of design review checklists for the Schematic Phase and Working Drawings Phase.

Project Name: **Date:**
Project Number:

A/E Checklist for HECO-4 Approval - Schematic Planning Documents

Design Narrative Information:	A/E	Proj. Mgr. spot check
Capacity and type of occupancy		
Functions to be housed in building		
Proposed Building location on site		
Exterior circulation-relation to other facilities		
Areas/capacities required for proposed activities		
Type of construction proposed and VUSBC number		
Outline description of basic materials		
Future construction or expansion, if any		
Desired style and character of building		
Structural design live, wind and earthquake load criteria		
Type of foundation system selected		
HVAC system type, est. heating/cooling loads and source		
Total square foot area per floor and building		
Total cubic foot volume		
Number of beds, seats or parking spaces where applicable		
Total estimated construction cost base on schem. Documents		
Total proposed project budget		

Drawing Requirements:		
Floor plans consisting of:		
Each floor layout		
Space names		
Nominal room sizes		
Circulation paths		
Roof Plan		
Longitudinal bldg. Section w/ floor to floor and floor to ceiling dim.		
Transverse building section		
Exterior elevation views		
Struct. Plan of typ. supported floor scheme		
Typical section showing proposed components of the floor syst.		
Orientation and approximate site plan locations of:		
Proposed roads		
Walks		
Parking		

Signed: ______________________________

Print Name: ___________________________

Title: ________________________________

This checklist is provided by Virginia Tech to the A/E for the purpose of documenting the A/E's quality assurance review in accordance with Section 8, Design Requirements and Standards, of the latest edition of the CPSM. Completion and certification of this checklist shall not relieve the A/E of full compliance with all laws, codes, regulations, policies and directives of the CPSM and VT Design Guidelines and Standards.

Rev. 01/06/04

Figure 3-14 - Design review checklist for the Schematic Phase

Project Name: **Date:**
Project Number:

A/E CHECKLIST FOR HECO-6 APPROVAL - Working Drawings & Specifications

	A/E	Proj.Mgr. spot check
Data Required:		
General and Front End Requirements per CPSM 808.1 and VT Design Guidelines		
Cost Estimate		
Permits and Utilities		
Calculations from each discipline involved		
Submittal Documents		
Working Drawings conforming to VT Drawing Standards as described below		
Drawing Requirements:		
Title Sheet:		
Project Identification-Agency, Proj. Code, Appr. Act #		
Location & Vicinity Maps		
Tabulation and floor area, total area & volume		
Tabulation of Units-Beds, seats or Parking spots		
List of Applicable Codes and Dates		
Building Purpose/Occupancy		
Use Groups per VUSBC		
VUSBC Construction Type		
Occupancy Loads per VUSBC		
Design Floor Live Loads		
Index of Drawings		
Site Plans (Comply with VT ***Site Development Specifications & Details*** where applicable)		
Based on approved Comprehensive Master Plan		
Scale and north arrow.		
New and existing contours affected by work.		
Floor and contour elevations.		
Applicable boundaries with survey computations.		
Dimensioned relationship of new work to boundaries and existing structures.		
Locations of test borings.		
General parking and handicap parking.		
Handicapped-accessible routes		
Pedestrian traffic routes.		
Demolitions: structures, walks, utilities, trees, etc.		
Proposed landscaping (planting materials selection per Office of University Architect)		
Existing and new utilities:		
storm sewers, sanitary sewers, water supply, gas, steam distribution pipes and tunnels, electric and telephone poles and lines, hydrant locations and data on fire flow test.		
Site improvements such as fencing, lighting, etc.		
Typical paving section for proposed types/thicknesses per VT Design Guidelines.		
Identify/show special earthwork recommended and construction considerations noted in soils report.		
Demolition drawings		
For interior demolition:		
Floor Plans showing existing, etc., and showing or describing existing materials/ construction to be removed		
Information or estimates for bidding for work to be removed		
For total building demolition:		
Plan of building with length and width dimension		

Figure 3-15 - Design review checklist for the Working Drawings Phase (only page 1 of 5 is provided for this example).

a) Programming Phase

This stage involves developing the conceptual layout, space adjacency relationships and building functionality. It is the responsibility of the architect to transfer University descriptive requirement into a building format.

The Programming document was prepared by the A/E. It was the Project Manager's responsibility to ensure the document was reviewed and approved before the following Schematic Phase can proceed.

b) Schematic Phase

This is the initial milestone of the drawing phase both in the design and review process. Generally, the Schematic Phase translates the written project functional, spatial and adjacency requirements into a more graphical representation of floor plans, space sizes (dimensions) and relationships, and the exterior building elevations.

Based on the `Basis of Design' for the schematic phase, the A/E prepares the schematic design. The `Basis of Design' is a narrative description of the project and is a bound presentation of facts sufficiently complete to expedite BCOM review of the Schematic submittals. The Schematic Basis of Design narrative presents the basic information, criteria, logic, evaluations and considerations developed in each category to prepare the Schematic submittal. Design computations, sizing of members, details of connections, etc. are not required to be submitted with the Schematic Basis of Design. However, general computations supporting system selection, member depths, floor-to-floor heights, mechanical and electrical loads should have been made.

The reviewers include CDCD, PP, Utilities, EHSS and CNS. The reviews were then passed back to the A/E through the PM, so comments and changes can be incorporated into the design. User representatives (occupants) then reviewed the schematic design and at the same time, the HECO-4 approval form was prepared. The HECO-4 form is a checklist used by the A/E during the schematic phase to ensure that all the required and related items are available and not overlooked. The designs had to be HECO-4, BOV and AARB approved before the next phase of design can take place.

c) Preliminary Phase

The next project approval milestone is the submittal of preliminary drawings with the basis of design for the preliminary phase, building systems and equipment checklist, and cost estimates. The Preliminary Basis of Design narrative expands on the Schematic submittal to reflect the further analyses, evaluations and selections/decisions made to arrive at the Preliminary level of design.

The A/E prepared and presented the preliminary design plans. These designs were reviewed again by the CNS, PP, Utilities, EHSS and CDCD. During this time the HECO-5 checklist and independent cost estimate was prepared and analyzed. The HECO-5 checklist outlines the preliminary design submittal requirements. The Hokie Stone requirements were also confirmed.

The reviewers' comments were transmitted through the PM, back to the A/E to incorporate any changes required. The completed preliminary designs were then transmitted to the DCR and AARB for review and approval. Together with HECO-5 checklist approval, the designs proceeded to the Working Drawings phase.

d) Working Drawings Phase

The next project approval milestone is the submittal of completed working drawings and specifications.

This phase started off with the estimate and approval for BAS (Building Automation System) procurement. Proprietary Purchases were requested and approved. The CNS and Utilities reviewers reviewed the working drawings and the related estimates. Reviews were also performed by CDCD, PP and EHSS. The DCR reviewed for soil and erosion. The cost estimate analysis was also undertaken for the whole project. The comments from the reviewers were then passed to the A/E. Any changes required and highlighted by the reviewers in the form of review comments were incorporated in the bid package before the construction documents are released to prospective bidders. At the same time, the approval from the Storm Management was also received.

The working drawings were then submitted to DEB for review and their comments (if any) were incorporated in the drawings. At the same time, the HECO-6 checklist was prepared and the project budget was aligned. The working drawings were now ready to go to the bid stage when HECO-6 was approved. HECO-6 ensures that before the release of the working drawings to the bidders; any Codes and Standards deficiencies are corrected.

Category 3: Design Review Information Reference Sources

The following references were used and complied with when creating the design documents and performing the design review.

a) Basis of Design

Basis of Design is the documentation of Virginia Tech's interpretation of the project requirements and where decisions that set the tone for the project are made. It is formally developed with the A/E on the A/E's selection as the principle designer for the project. Basis of Design for the schematic phase, the A/E prepares the schematic design. The 'Basis of Design' is a narrative description of the project and is a bound presentation of facts sufficiently complete to speed up BCOM review of the Schematic submittals. The Schematic Basis of Design narrative presents the basic information, criteria, logic, evaluations and considerations developed in each category to prepare the Schematic submittal. Design computations, sizing of members, details of connections, etc., are not required to be submitted with the Schematic Basis of Design. General computations supporting system selection, member depths, floor to floor heights, mechanical and electrical loads should be made. The Preliminary Basis of Design narrative expands upon the Schematic submittal to reflect the further analyses, evaluations and selections/decisions made to arrive at the Preliminary level of design.

Basis of Design document generally consists of these sections:

1) Architectural Design Criteria which outlines the Building Use and Construction Classification and the Facility Area Summary
2) Structural and Civil Engineering Design Criteria
3) Mechanical & Plumbing Systems Design Criteria
4) Power, Lighting, and Data Systems Design Criteria
5) Outline Specifications (VT Design Guidelines and Addenda)
6) Project Cost Estimate

b) Drawings

2D CAD drawings in forms of electronic in form of either AutoCAD *.dwg, AutoCAD *.dwf file format, and paper-based copies are used by design review entities throughout the design/design review phases. As time progresses, more detailed designs are produced to accommodate the project's requirement. For each stage by following a standardized checklist, design reviewers review the 2D drawings and then typed-in their comments in a word document or form. A/E will review the comments and if changes are needed, they will be incorporated in the design or design documents.

c) Specifications

The Virginia's Construction and Professional Services Manual (CPSM) defines Specifications as classified into 3 types:

1) Non-proprietary and Performance Specifications: This is a method of specifying materials, equipment and systems. A non-proprietary specification is written either as (a) a generic performance specification or as (b) a specification naming a minimum of three manufacturers with model or series numbers.

2) Proprietary Specifications: A specification is proprietary if it specifies a product/requirements which only one manufacturer can meet but the product is available from multiple vendors or sources.

3) Sole Source Specifications: A specification is sole source when it names only one manufacturer or product to exclude others, or when it is contrived so only one manufacturer, product, or supplier can satisfy the specification. Because it eliminates competition, it can be used only in the most exceptional circumstances and under the strictest conditions.

d) Codes

Virginia Tech Design Guidelines and Construction Standards

(http://www.cdcd.vt.edu/Standards_Codes/Design_Guidelines.pdf)

This manual is used to guide and assist architectural and engineering consultants in the planning and preparation of design documents for construction and renovation of the university facilities. The guidelines in this publication identify specific or unique standards and requirements for University projects. The guidelines are intended to supplement the required codes, industry standards and other directives (CPSM and HECO Manual) which govern capital outlay work at Virginia Tech.

The Virginia Uniform Statewide Building Code (USBC) 2000 Edition and 2003 Supplementary Edition

(http://www.cdcd.vt.edu/Standards_Codes/2000%20USBC.pdf)

The VUSBC contains the building regulations that must be complied with when constructing a new building or structure or an addition to an existing building. The VUSBC's regulations also involve maintaining or repairing of an existing building, or renovating or changing the use of a building or structure in the state of Virginia.

Construction and Professional Services Manual (CPSM)

(http://bcom.dgs.virginia.gov/RDetailPg.aspx?I_PAGE_ID=4)

The CPSM sets forth the standards, policies, terms, conditions, and procedures to be followed by all departments, agencies, and institutions of the Commonwealth of Virginia in procuring professional services, designs and constructions of all structures (except roads and bridges which are under the purview of the Virginia Department of Transportation) which are on state property to include new construction, and renovations, modifications and additions to existing facilities.

The technical standards set in this Manual establish the levels of design, quality, energy efficiency, and performance required for projects on state property in addition to the minimum standards required by the applicable codes and standards for the project. These standards are intended to assure the protection of the public health, safety, welfare and accessibility as well as the protection of real property insofar as the use and occupancy of buildings on state property are concerned. The A/E has to adhere to a checklist for each design phase which is found in "Appendix D" of the CPSM.

Higher Education Capital Outlay (HECO) Manual

The HECO Manual 2000 parallels the CPSM as part of a legislative action intended to decentralize design and construction processing, enabling qualified higher education agencies of the Commonwealth of Virginia, such as the Virginia Tech, to establish design and construction requirements. On all phases of design, Virginia Tech provides HECO checklists to the A/E for the purpose of documenting the A/E's quality assurance review based on the CPSM. Completion and certification of the checklist does not relieve the A/E of full compliance with all laws, codes, regulations, policies and directives of the CPSM and VT Design Guidelines and Standards.

Other Codes

Discipline-based Codes, each has various related codes:

- Structural and Civil Engineering Design
- Architectural Design
- Mechanical & Plumbing Systems Design
- Power, Lighting, and Data Systems Design

e) Lessons Learned

In the design review documents, the author extracted comments that were made by the reviewers which refer to similar events that happened in previous projects. The author found that these comments and the information they carry should be captured and categorized under a category called “lessons learned”. Lessons learned can provide an opportunity for team members and stakeholders to discuss:

- successes that happened during or because of the project
- unintended outcomes that happened during or because of the project
- other things that in retrospect might have been better handled if done differently
- recommendations to others who might be involved in similar future projects

Shown below is one example of lessons learned. Other examples of Lessons Learned can be found in Appendix 2.

Lessons Learned 1
Subject: Wall construction and moisture problem
Project name: Career Services Building
Design Phase: Schematic Drawing
Discipline: Civil/Architectural/Structural & Building Code / Fire Protection
Reference: Wall sections

Reviewer’s comment:

Reviewer (a):

- Was concerned on the stability of the structure.
- Insisted that the Metal Stud Backup/GWB (gypsum wall board)/drywall backing were not as stable as block or brick veneer.
- The use of wall ties is a critical part of the assembly; Wall ties, Hokie Stone and GWB will not work well.

Reviewer (b):

- Was concerned about the similar issue as (a).
- Raised the issue of moisture migration problems in the past with Hokie stone, he was certain that such problem will not occur with CMU back-up wall.
- Agree with the use of GWB only if it was water resistant.

Category 4: Design Review Comments Classification

Figure 3-16 shows the ‘design review comments’ category that emerged from conducting Content Analysis on the three sets of Virginia Tech’s design review documents. The review comments can be classified into 2 main sub-categories; Error and Recommendation. Under each sub-category there are sub-sub-categories of comments. Examples based on these categories are provided from each project and can be found in Appendix 3.

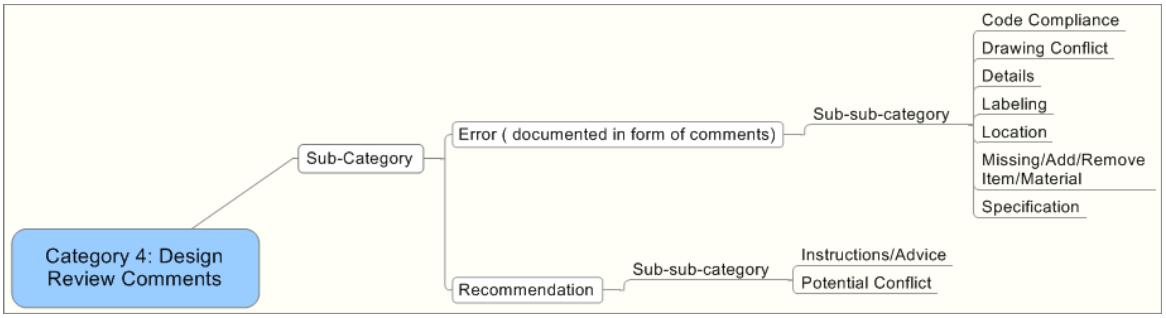


Figure 3-16 - Category 4 – Design review comments classification showing sub-categories and sub-sub-categories derived from Content Analysis of Design Review Documents (Virginia Tech's CSF, ACECHC & BP1 Projects)

Table 3-5 below shows a general description of each sub-sub-category.

Sub-category - Error	
Sub-sub-category	**General description of the sub-sub-category**
1. Code Compliance	This occurs when reviewer reminds/highlights designer about certain codes that need to be complied with. E.g. Career Services: The designer is reminded to check the specifications in all roofing sections against Section 707B of the CPSM. Reviewer highlighted certain codes, a) Provisions pertaining to roof inspector should comply with Section 707B.5. b) Section 707B.6.2 specifies the Pre-Roofing Conference should take place before materials are ordered. c) Comply with CPSM Section 707B.14.6 regarding roof protection. d) Comply with CPSM Section 707B.14.7 regarding final inspection procedures.
2. Drawing Conflict	This occurs when information in two or more drawings which should synchronize or coordinate, conflict each other. E.g. Career Services: Reviewer points out three related sheets on Scupper Detail and Wall Section details. The conflict is about the height of parapet wall in scupper details did not match parapet wall on exterior elevations.

3. Details	This occurs when information in detail drawings such as wall section, stairs, etc is incorrect or incomplete. E.g. Career Services: In reference to Drawing A604 Interior Stair and Rail Details, designer is asked to include a Plan/Section of Post shown on Details 1, 2, 13, 14, and 15/A604.
4. Labeling	This occurs when changes required for labeling or wording without the need to major redesign. E.g. Career Services: In reference to Sheet C-2. Existing Conditions and Demolition Plan utilizes the seal of Dan Huff, Professional Engineer in the Commonwealth of Virginia, who is a part of Draper Aden Associates. However, Sheet C1 notes the survey of the project site was performed by Anderson & Associates Incorporated. Designer is asked to confirm and indicate on the drawings that the survey is the responsibility of Draper Aden Associates, who is subcontracted to Hanbury Evans Wright Vlattas & Co (HEWV).
5. Location	This occurs when there is the need to relocate or change of locations of items such as positioning of doors, fixtures etc. E.g. Career Services: In reference to Elevator Machine Room. Designer is asked to relocate elevator machine room to area adjacent to elevator on first floor. Also pre-action system is to be moved to room 120, and floor drain is to be added in 120.
6. Missing/Add/Remove Item/Information	This occur when items or information were found missing, need to be added, need to be removed, or reviewer needs clarification. E.g. Career services: In reference to Interior Stair and Rail Details Drawing A604. Designer is to include a Plan/Section of Post shown on Details 1, 2, 13, 14, and 15/A604.
7. Specification	This occurs when item in drawings need to strictly follow certain specifications. E.g. Career Services: • Specifications, Section 16500-2.01.B.1: Fluorescent lamps shall be low mercury content

	type. • Specifications, Section 14240-2.2.B: Architect should select option 2 as preferred standby powered lowering operation. • Specifications, Section 14240: Provide PVC jacket for jack cylinder.
Sub-category - Recommendation	
Sub-sub-category	**General description of the sub-sub-category**
1. Instruction/Advice	This occurs when reviewer provides instruction, strong recommendation, or personal opinion based on his/her knowledge and experience. This piece of information or knowledge that may not be found in any of the existing documentation and drawings. E.g. Career Services: In reference to Drawing S102, Note 'S' Elevator Hoist Beam. According to reviewer "Do not remove hoist beam after elevator is installed".
2. Potential Conflict	This occurs when reviewer foresees that conflict is highly probable, thus he/she expresses concern. E.g. Career Services: Reviewer expresses his concerns as follows, "There needs to be a utility corridor established for this site. Fire and domestic water, chilled water, hot water and sanitary sewer are going to be routed along the north side of Career Services and Student Services. There is a new dorm planned for the area to the north of these buildings. The routing of these utilities needs to be carefully planned to avoid future conflicts. Some of this work needs to be done in the near future because of the start of construction on Student Services. As a minimum, the Career Services design team needs to provide VT with a site plan showing the definitive location of Career Services so proper routes for the utilities can be established."

Table 3-5 - A general description of each sub-sub-category

Besides performing the content analysis on the design review documents, the author also determined and then tabulated the number of occurrences of the types of design review comments based on the categories and sub-sub-categories. Table 3-6 shows a summary of the number of occurrence of each type of the comment derived from the content analysis of the design review documents.

Table 3-6 evidently shows that a larger project i.e. Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech Complex has more design errors compared to the smaller project i.e. Career Services Facility Building. Larger projects usually carry larger amount of design and documentation, therefore design errors are more typical.

Table 3-6 also reveals there is no consistency pattern or trend which type of error that usually occurs in design. This fits perfectly with producing design activities involve various uncontrollable variables such as “different designers for each project, “who designed it”, “the experience of the designer”, “accuracy of interpretation of the owner’s design requirements” etc.

Type of Comments	**Career Services Facility Building**	**Bioinformatics Phase 1**	**Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech Complex**
Code Compliance	13	144	102
Drawing Conflict	9	14	49
Details	48	36	73
Labeling	27	76	85
Location	22	13	40
Missing/Add/Remove Item/Information	63	203	271
Specification	27	34	47
Instruction/Advice	83	215	244
Potential Conflict	14	26	47
Total	**306**	**761**	**958**

Table 3-6 - The number of occurrence of each type of comment from the content analysis of the design review documents

3.3.4 Dissemination of Work (Publications in Conference Proceedings and Journals)

The author with assistance from his advisor disseminates the author's progressive findings in conferences, conference proceedings, and journal publications (Table 3-7). The dissemination of the author's work helped him to identify, promote and confirm the problems, and promote the potential solutions.

P #	Year	Chapter	Title	Conference Proceedings or Journal
		Literature Review		
1	2001	Virtual Environment/VE in General	Making the Transition toward an Alternative VE.	Proceedings of the AVE II & CONVE 2001 - Conference on Applied Virtual Reality in Engineering & Construction Applications of Virtual Reality: Current Initiatives and Future Challenges, CHALMERS University of Technology Goteborg, Sweden, 14-15 Oct, 2001.
2	2001	Information visualization in construction project	Visualizing Knowledge In Construction Project : The Macro Planning Stage	Proceedings of Knowledge Management International Conference and Exhibition 2001 (KMICE 2001), Langkawi, Malaysia.
3	2002	VE in Construction	Virtual Reality in Construction	Engineering Computational Technology, Saxe-Coburg Publications, pg. 25 – 52 (Chapter Contributor), Saxe-Coburg Publications, 2002: ISBN 1-874672-17-2.
4	2004	VE/3D Visualization in Construction	3D Visualization to Facilitate Understanding of Assemblies and Details in Construction Education,	Proceedings of the American Society for Engineering Education (ASEE-SE Section) Annual Conference, Educating Engineers for the Information Age, Auburn University, April 2004.
5	2005	VE/3D visualization in Construction Education	Interactive 3D Visualization as a Tool for Construction Education	The 6th International Conference on Information Technology Based Higher Education and Training (ITHET 2005), Santo Domingo, Dominican Republic, July 7-9, 2005.
		Data Analysis		
6	2003	Correlation between VE technology and slow adoption	Issues in Implementing a Virtual Environment based Design Review System	Proceedings of the Conference on Construction Applications of Virtual Reality (CONVE 2003), Virginia, USA, September 24 – 26, 2003.

		Framework	***For this section of the thesis, the author documented the why and how VE, IS, DB, context-aware information processing, 3D Game Engine are incorporated for design review purposes***	
7	2002	Prototype Development: General implementation issues	Virtual Office Walkthrough Using a 3D Game Engine, Special Issue on Designing Virtual Worlds	International Journal of Design Computing (IJDC), Vol 4, 2002
8	2002	Prototype Development: Game Engine Technology issues	Remote Collaborative Virtual Walkthroughs Utilizing 3D Game Technology	Proceedings of the 2002 ECPPM: eWork & eBusiness in AEC, Portorož, Slovenia, Sept 9-11, 2002.
9	2002	VE Display and user interface	3D Visualization Using the Pocket PC	Proceedings of the 2002 ECPPM: eWork and eBusiness in AEC, Portorož, Slovenia, September 9-11, 2002.
10	2003	Prototype Development: General implementation issues	Implementation Issues of a Design Review System using Virtual Environment	Proceedings of the ASCE Construction Research Council, PhD Research Symposium, November 14, 2003, Nashville, Tennessee, USA.
11	2003	Prototype Development: Collaboration issues	A Framework for a Collaborative Design Review System Utilizing the Unreal Tournament (UT) Game Development Tool	Proceedings of the 20th CIB W78 Conference on Information Technology in Construction Waiheke Island, Auckland, New Zealand, 23-25 April 2003.
12	2004	VE Display and user interface	Evaluating the Effectiveness of Virtual Environment Displays for Displaying 3D Models and Tasks in Construction	Proceedings of the Conference on Construction Applications of Virtual Reality (CONVE 2004), ADETTI/ISCTE, Lisbon, 13-15 Sept 2004.
13	2007	Context-aware design review information processing	Information Rich Virtual Design Review Environment	Proceedings of the 24th W78 Conference, Maribor 2007 & 5th ITCEDU Workshop & 14th EG-ICE Workshop, 26 -29 June 2007.

Table 3-7 - Dissemination of work

3.4 Conclusion

The literature review and case study (consist of interview and content analysis of design review documents) undertaken for the research revealed the construction industry (in general), and in carrying out design review (in particular) are facing the following challenges:

- Design review information is "scattered" thus creating problems for design reviewers to access particular information at the right time, at the right place, and to and from the right person or source. The author lists ten issues that fall under this category. Literature review, interviews and content analysis of design review documents revealed similar results on this issue. Interview specifically drew attention to issues on the need for better methods of accessing relevant design review information. All the companies agreed that design review information is too scattered all across volumes of manuals.

- The imperfect or "flawed" representation of 2D designs has been the medium that is widely used in communicating designs among project teams and conveying owner's design intentions. 2D representations do not represent satisfactory spatial information. Everyone involved is aware of the problem, but little move is made to transition to 3D design representations. Literature review and interviews revealed similar results on this issue.

  3D design representations via 3D models may improve 2D problems, but still without the needed information, a 3D representation will be less useful. More could be done to make the 3D model intelligent and more useful. Interview revealed main suggestions by interviewee on 3D models and design review. Interviewees suggested that:
    a) The use of 3D model should start from the beginning.
    b) The 3D model must also be able to produce 2D plans for construction documents.
    c) Using 3D model is a better way to communicate, collaborate, and perhaps sharing and viewing the design in real-time. One company suggested videoconferencing while another used web-conferencing service (through Microsoft Netmeeting).

- The current design review approach is closely tied to the sequential stages of design. Literature review, interviews, and content analysis of design review documents revealed similar results on this issue. Design is passed to and from designers to design reviewers and sometimes a project manager following i.e. programming, schematics, preliminary, and working drawings phases. The next phase can only begin if the previous phase is satisfactory and complete. This process is usually iterative and undergoes "hick-ups". This can be improved and time can be saved if all designers, reviewers and project team can review design simultaneously and communicate in a shared space.

- The construction industry is slow to adopt new technologies. VE is one of the state-of-the-art 3D representation technologies which are underutilized by the industry. Literature review and interviews revealed similar results regarding this issue. Even though much research has been done, most application do not spread openly into and available "cheaply" to the construction community. A 3D VE, reviewers can visualize and interact with the virtual objects, and unlimited virtual walkthroughs of the facility can be made.

The content analysis on design review documents of Virginia Tech's past projects has accomplished in extracting the main categories or classifications of design review information. The four main categories are: Design Review Entity, Phases of Design Review Process, Design Review Information Reference Sources, and Categories of Design Review Comments. Under these categories, further categorization was made i.e. sub-categories and sub-sub-categories. The author used these findings to formulate a framework for a context-aware processing of the required design review information in a VE that employs a context-aware concept.

CHAPTER 4

4 Framework for Context-Aware Information Processing for Design Review in a Virtual Environment

4.1 Introduction

This chapter describes the framework for context-aware processing of design review information in a VE. The framework incorporates the use of information filtering using context-aware concepts. The author uses the Torque Game Engine (TGE) as the enabling VE development tool.

4.2 Components of the Framework

The components of the framework for context-aware information processing for design review in a VE are (Figure 4-1):

1) Context-aware information processing, and
2) the VE

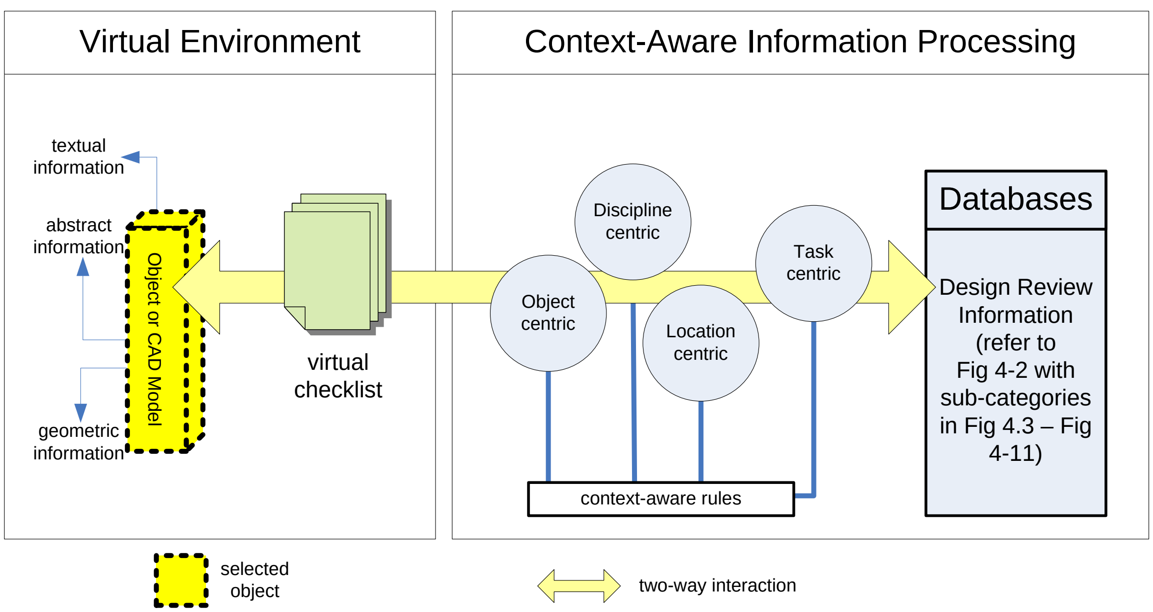


Figure 4-1 - A diagram showing the framework for context-aware information processing for design review in a VE

4.2.1 Context-Aware Information Processing

As shown in Figure 4-1, the context-aware information processing component is responsible for filtering, retrieving and storing the design review information during a design review session. This component consists of:

1) a collection of databases where design review information is stored, and
2) the context-aware rules that accommodate the information processing of the design review information stored in databases

4.2.1.1 Databases for Design Review Information

Prior to the creation of the databases, categories of design review information are established. These categories are determined based on the findings from the Content Analysis of design review documents of Virginia Tech projects (refer to Chapter 3: Section 3.3.3). To construct these categories, the author synthesized (combined) the information from the analysis with new information and its attributes, that are required to serve the needs for design review in a VE (e.g. information and its attributes for 3D objects/model, information stored as lessons learned, information gathered to develop virtual checklists etc.).

Through the synthesis, some elements from the Content Analysis in Chapter 3 were discarded because of redundancy and irrelevance to the new design review approach, such as the use of multiple sets of disciplinary-based 2D drawings. By using a 3D model, designers from each discipline embed their design work into the 3D model. Any errors or deficiencies are thus visible on the 3D model and can be corrected early on. Also, more significantly, the common problems of coordination errors or conflicts in drawing can now be reduced, or even eliminated.

Figure 4-2Figure 4-2 shows a summary of the categories of design review information that has been synthesized for the design review framework.

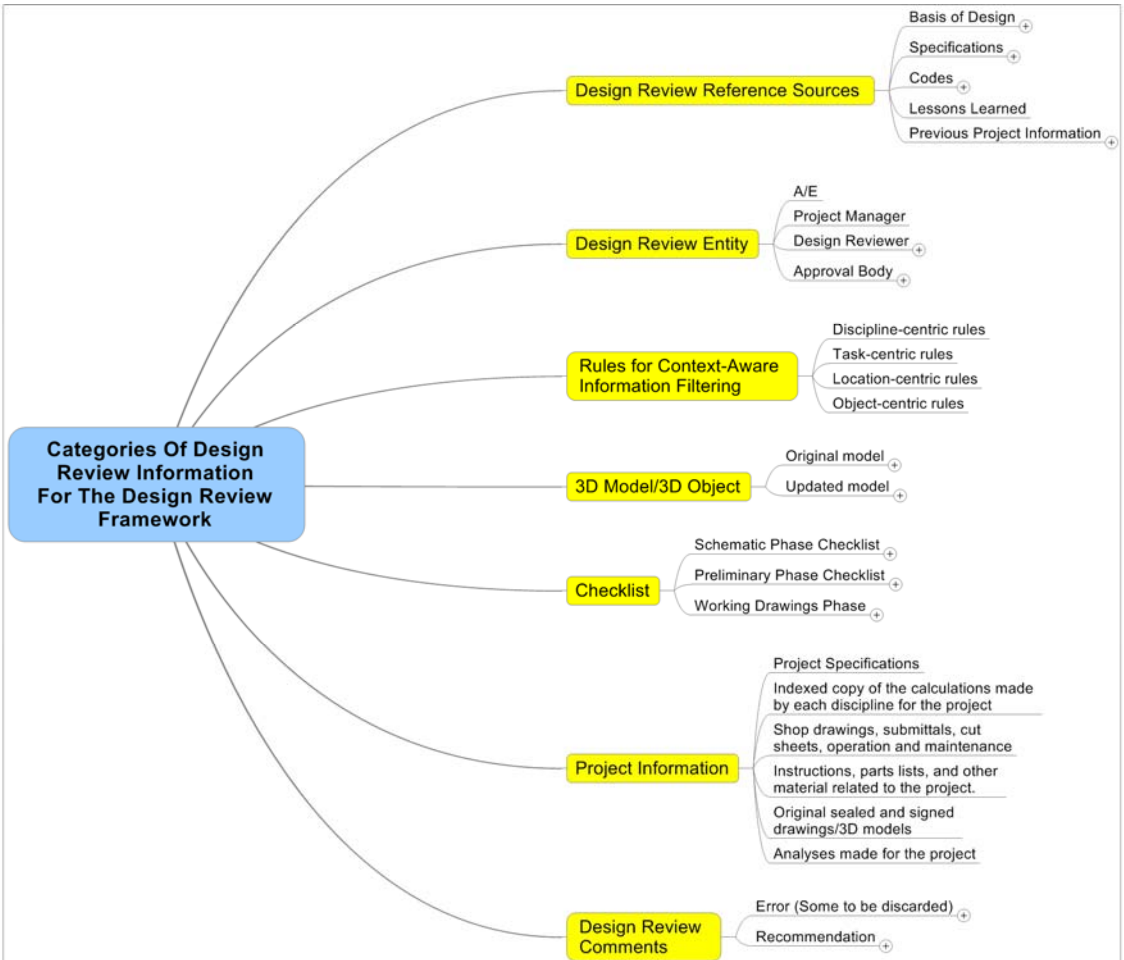


Figure 4-2 - Summary of the categories of design review information synthesized for the design review framework

Figure 4-2Figure 4-2 describes the following:

- Information is divided into seven major categories i.e. Design Review Entity, Project Information, 3D Model/3D Object, Rules for Context-Aware Information Filtering, Checklist, Design Review Reference Sources, and Design Review Comments and Recommendations.
- These categories also represent the major databases that are part of the application. As each category is a database, each database contains collections of data and information.
- Each category can be further expanded into smaller categories and into detailed information i.e. attributes. Since these categories represent databases, the information in these categories can be linked to one another by establishing relationships among information.

An example of a piece of information which is expanded in detail is shown in Figure 4-3 (refer to Appendix 4 for a larger visual representation). The category of information was extracted from Virginia Tech's past projects, and then put in place with newer category of information to support design review in a VE.

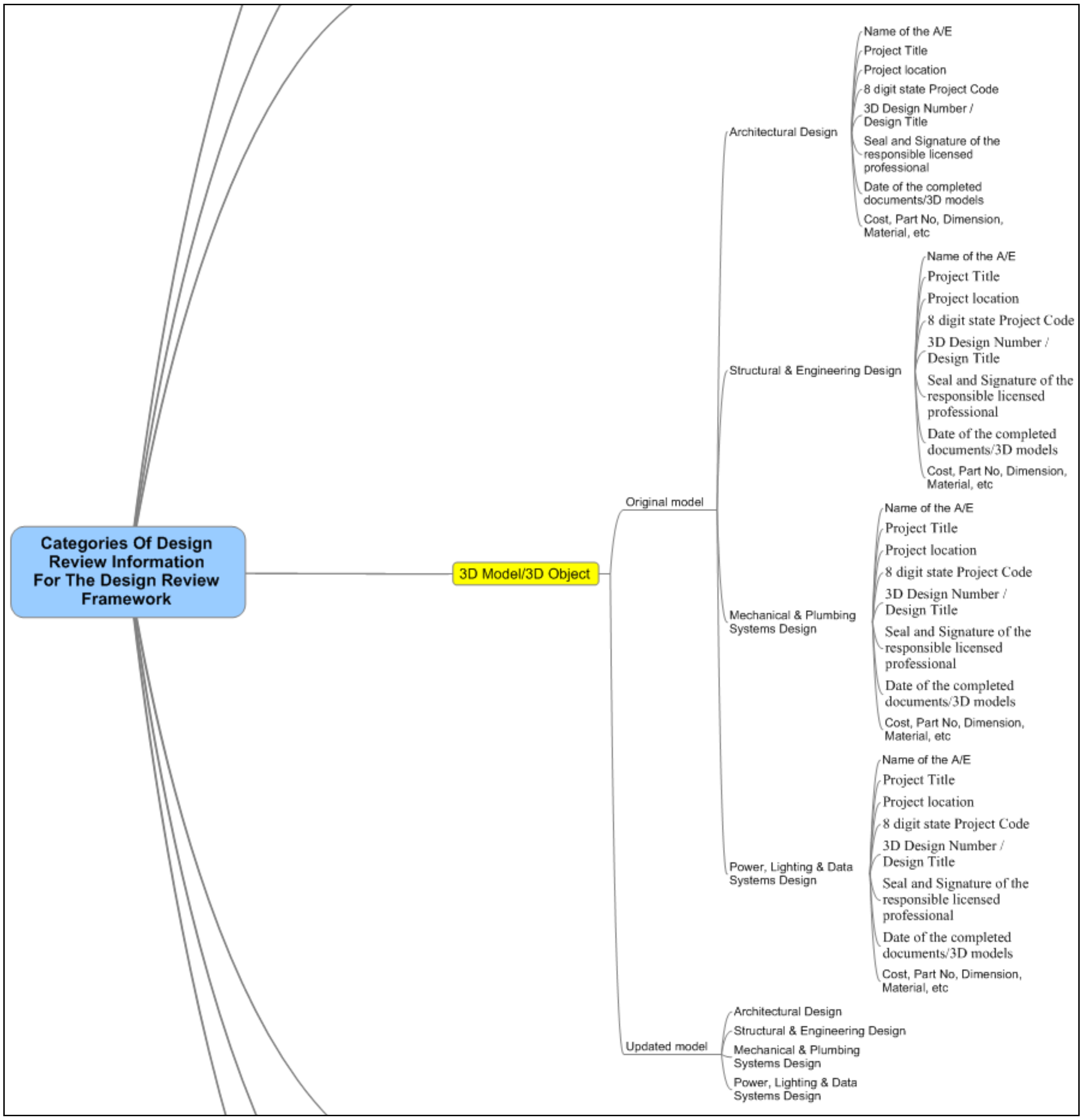


Figure 4-3 - An example showing the 3D model category being expanded

The databases are the repository where design review information is collected, organized and stored for use and retrieval during design review. Some of these databases are linked to the 3D models database. The followings are the information located in the databases.

a) Design Review Entities

Design review entities are: Designers (A/E), Project Manager, Reviewer, and Approval Bodies (Figure 4-4).

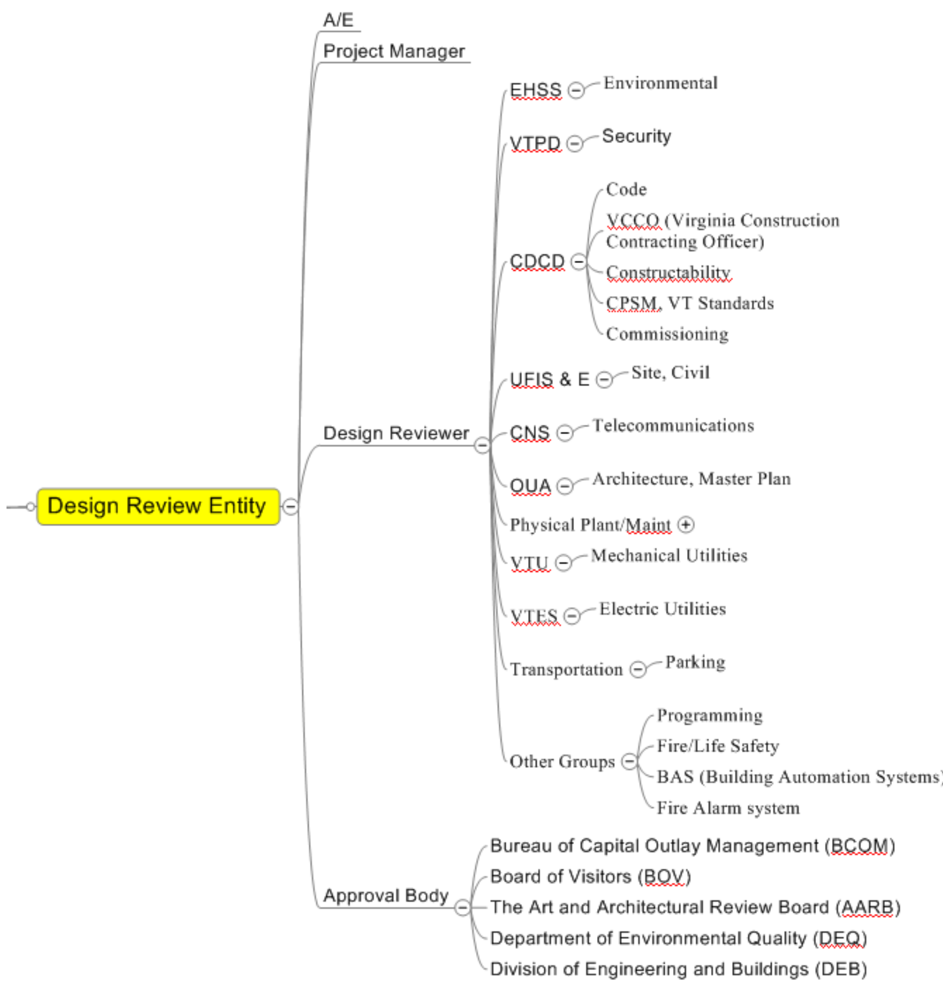


Figure 4-4 - Design review entities

Examples of the attributes for a reviewer in the database are:

Name_Reviewer
Affiliation_Reviewer
Reviewer_ID
Project_ID
Design_phase_ID
Review_date
3D_design_number
Recommendation_number
Comment_number

b) *Project Information*

This information is continuously updated as design work, design review and design modification progresses. The final information is used for final bid package. This database also stores previous project information. The sub-categories of the project information are shown in Figure 4-5.

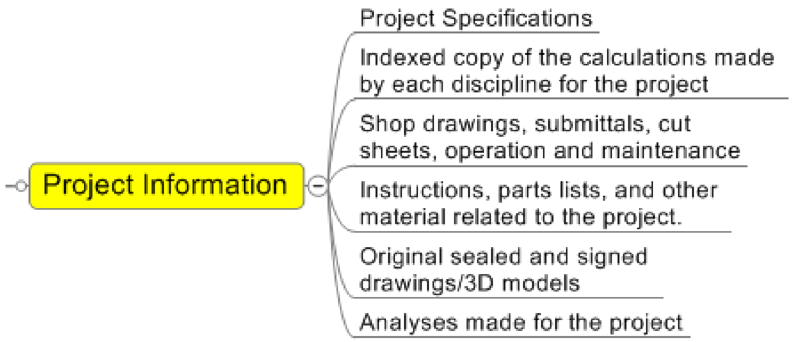


Figure 4-5 - Categories of the project information

Examples of attributes stored in this database are:

Project_title
Project_location
State_Project_Code
Designer_ID
Affiliation_Designer
3D_design_number
Date_complete
Date_update
Specification_ID

c) *Design Review Reference Sources*

This database stores reference sources that are usually used and complied with by design reviewers and designers when creating the design documents and performing the design review. These sources are (also shown in Figure 4-6):

- Basis of Design
- Specifications
- Lessons Learned obtained from current and past projects
- Codes - Building Codes from USBC (Uniform State-wide Building Codes), IBC (International Building Codes), National Fire Prevention Association Code 1996
- Manual and Standards – ASHRAE (American Society of Heating, Refrigerating and Air-Conditioning Engineers) Handbooks, ASTM (American Society for Testing and Materials) Standards, AISC (American Institute of Steel Construction) Manual.

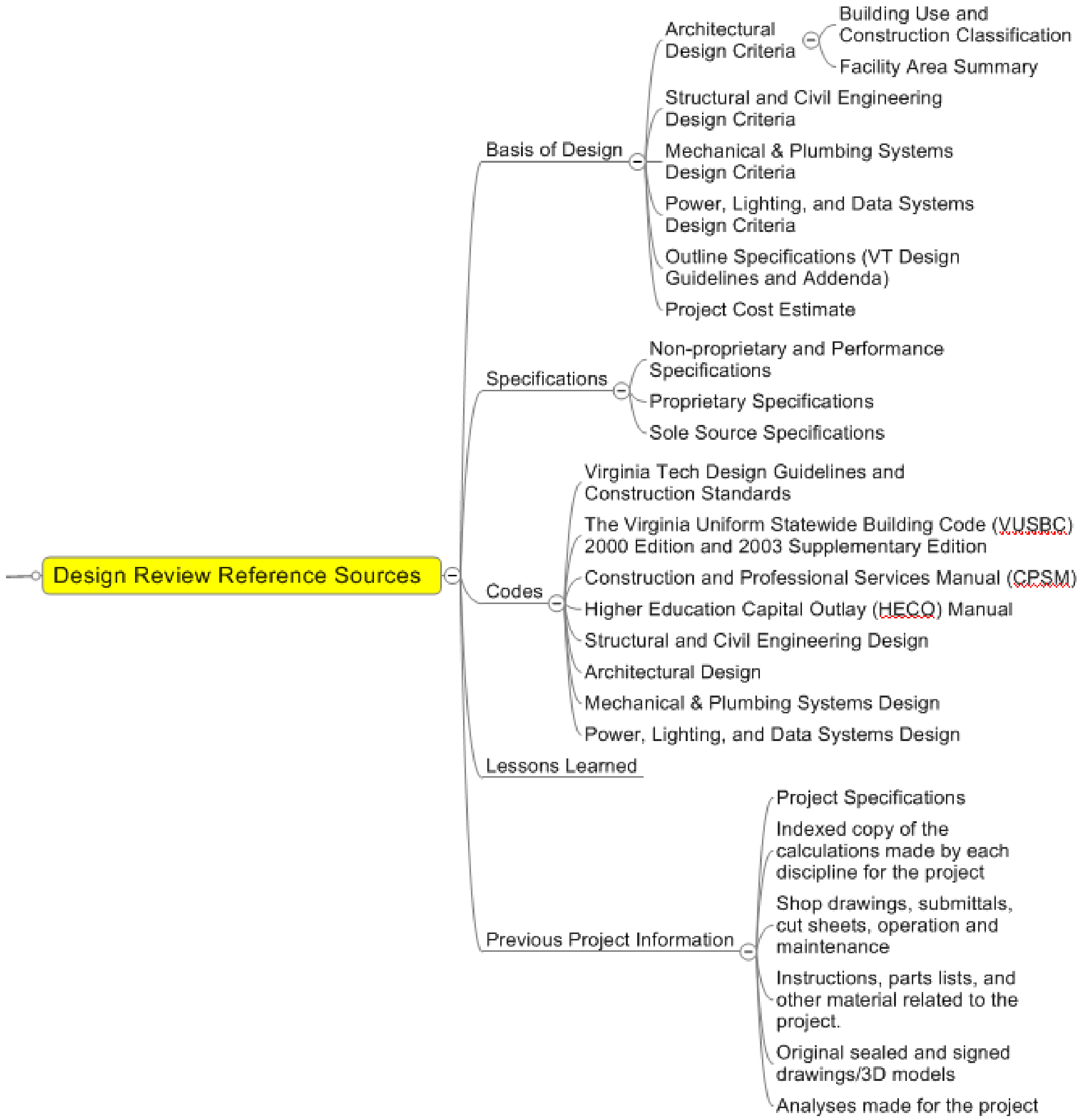


Figure 4-6 - Design review reference sources

Examples of the attributes in this database are:

Basis_of_Design_ID
Project_specification_ID
Lessons_Learned_ID
USBC_Code_number
IBC_Code_number
ASHRAE_Standard_number
NFPA_Code_number
ASTM_Standard_number
AISC_Specification_number

d) *Checklists*

This is the database for checklists for building systems that design reviewers use: Architectural Design, Structural and Engineering Design, Mechanical, Electrical and Plumbing Systems Design, and Power, Lighting and Data Systems Design. The checklists help guide the design review process and ensure potential design issues can be avoided. Specific Checklists are used for specific phase of design i.e. Schematic Design Checklist, Preliminary Design Checklist, and Working Drawings Checklist (Figure 4-7).

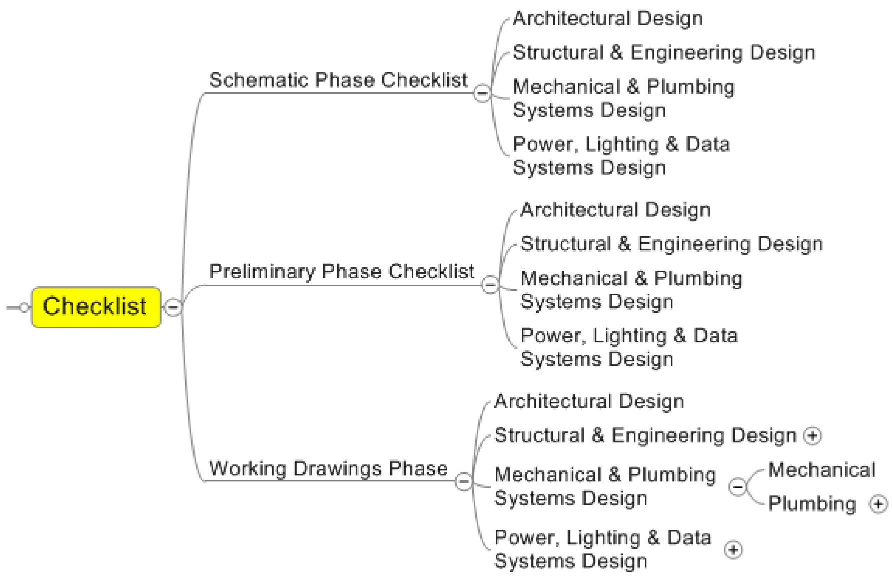


Figure 4-7 - Design review checklists

The following example of attributes relate to the checklist for the structural review of roofing system:

Project_title
Reviewer_ID
Location_ID
Material_specification
Dimension_specification
Live_load
Wind_load
Seismic_criteria
Comment_number

e) Design Review Comments

This is the database for comments and recommendations that are created by design reviewers. Comments are created when a reviewer finds an error or inconsistency in design and then alert the designer by creating a comment input in the comment form. Recommendations are created when a reviewer has a recommendation to make, to correct an error or inconsistency. Recommendations are also created when reviewer has suggestions or better ways to implement something, or when reviewer found potential conflicts or problems that might occur on site due to inconsistencies in the design. Specific types of errors are: Code Compliance, 3D Design Conflict, Details, Labeling, Location, Missing or Add or Remove Item/Information and Specifications (Figure 4-8).

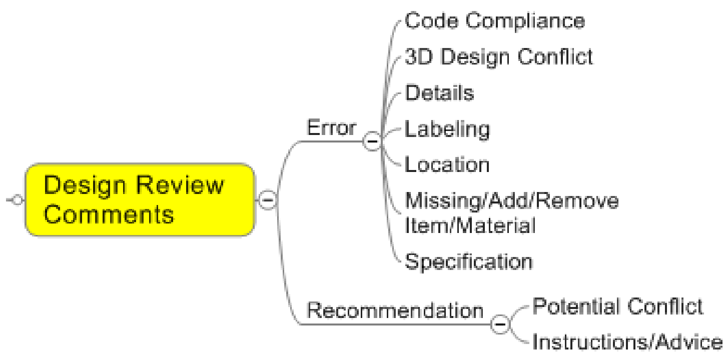


Figure 4-8 - Design review comments

Examples of the attributes in this database are:

Name_Reviewer
Reviewer_ID
Project_ID
Design_phase_ID
Review_date
3D_design_number
Error_type
Recommendation_number
Comment_number

f) *3D Model/3D Object*

This database stores the information (abstract, geometric and textual) for the 3D Model and 3D object (Figure 4-9).

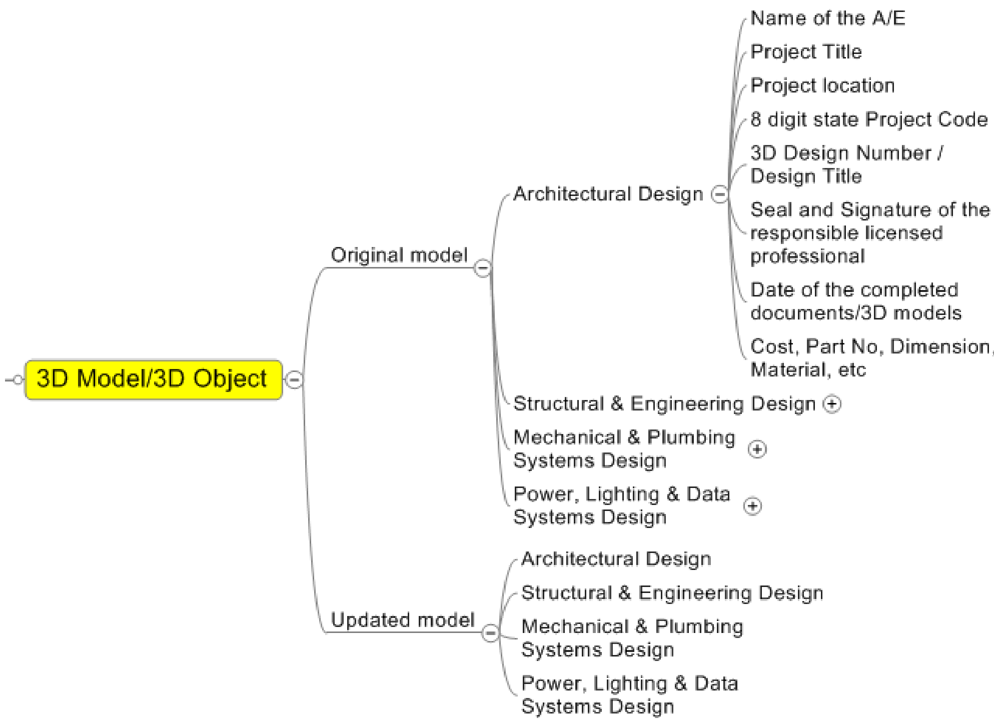


Figure 4-9 - Information for the 3D model and 3D object

The following example of attributes of 3D Model/3D object relate to the fixtures under Electrical System.

3D_design_number
Project_ID
Reviewer_ID
Design_phase_ID
Review_date
Fixture_location
Material_specification
IFC_Code_number
Comment_number

g) *Rules for Context-Aware Information Filtering*

This is the database that stores the all the rules for the design review information filtering which is based on context-aware concepts. The four contexts are discipline-centric, task-centric, location-centric and object-centric (Figure 4-10).

Figure 4-10 - Rules for context-aware information filtering

4.2.1.2 Context-Aware Information Filtering

The role of information filtering is to remove irrelevant information by bringing only the relevant information to the design reviewer's attention. Whenever a design reviewer needs specific design review information, he "requests" the information and the filtering process begins. This process calls for the use of rules that reside in the rules database. Once the specific information is found, the information is then passed back and visually displayed to the design reviewer in the VE.

The rules overcome the problem of manually processing the multitude of design review information, and facilitate information processing i.e. filter and retrieve. The rules bind and guide the processing of design review information. The rules are based on the questions that occurred in real-world during design review. In practice, a reviewer will have two questions in mind: (#1) who am I? and (#2) what do I want to review? When these two questions have been answered, the rules sort out and anticipate the information required by the reviewer. Sub-questions following question (#2) that a reviewer may ask himself can be (not in any specific order); (#2a) where am I? (#2b) what construction component do I want to review? (#2c) what do I want to do? These rules assist the user in filtering, querying, sorting and displaying data and information while viewing the 3D model in the VE.

The framework supports four contexts of filtering information before providing and displaying the relevant information to reviewers in the VE. These contexts are:

1) a discipline centric context (D): based on the type/role of reviewer,
2) a task centric context (T): based on the tasks and objectives of the design review session being performed,
3) an object centric context (O): based on the graphical component/assembly selected by the reviewer,
4) a location centric context (L): based on the relative spatial position of the reviewer within the 3D model in the VE.

The four contexts described above are based upon the concept of context-aware computing. Context-aware concept helps in the implementation of the filtering mechanism of design review information. Context-aware computing is an extended form of mobile computing. Schilit *et al.* (1994) states that the concept of context-aware uses the *mobile distributed computing system* because it concerns not just mobile computers but also mobile people. Context-aware computing spans a multitude of situations and locations, unlike desktop computing that occur at a single location. Users can access resources from wireless portable machines or stationary devices, and computers connected to local area networks. For this research, design reviewers can continuously be on the move (or virtually moving) in the VE, and at the same time able to access design information resources.

Context-aware computing concepts are also useful as it provides “situational information” for design reviewers. Suchman (1987) acknowledged that “situational information” is handy for users since human activity is very “situation dependent” and does not always go according to plan. Dey (2001) described “situational information” as context. Dey specifically defines context as “any information that can be used to characterize the situation of entities”. Dey elaborates this definition as “typically the location, identity, and state of people, groups, and computational and physical objects.” The four contexts developed in this research are based upon Dey’s (2001) description of context.

Elsewhere, Schilit *et al*. (1994) observed, “Context encompasses more than just the user’s location, because other things of interest are also mobile and changing. Context includes lighting, noise level, network connectivity, communication costs, communication bandwidth, and even the social situation; e.g., whether you are with your manager or with a co-worker.” The four contexts developed for this research also refer to Weiser (1991) i.e. “Context-aware applications need contextual information to deliver the correct service to the correct user, at the correct place and time, and in the correct format for the environment.”

The goal of context-aware computing is to acquire and use context from a computer device in order to provide appropriate services given the acquired context. Schilit *et al.* (1994) described context-aware computing software as “software that examines and reacts to an individual’s changing context.” Satyanarayanan (2002) extended the description of the software as “…aware of its user’s state and surroundings, and help it adapts its behavior.” Schilit *et al.* (1994) described four categories of context-aware computing concepts: proximate selection and contextual information, contextual commands, automatic contextual reconfiguration, and context-triggered actions.

Specifically, the design review framework developed by the author falls under the category of Context-Triggered Actions (Schilit *et al.*, 1994). Under this category, the system uses simple IF-THEN rules to trigger or specify how the design review application should adapt and present specific information for a design reviewer during design review in the VE.

a) Interaction among Contexts

Designations are assigned to each context. T represents the task that will be performed, O being the object of interest, L corresponds to the location of the reviewer within the 3D VE, and D represents the different construction disciplines which includes architectural, structural, civil, mechanical, electrical and plumbing. Each entity within a context is given a designation for creating the relationships between the disciplines D, the objects O, the locations L, the tasks T and the information stored in the database. New designations can be created and assigned should there arise any needs for them. The designation to the different entities in each context is denoted by alphanumeric subscripts. Table 4-1 shows some examples of the designation.

Designation	***Contexts***	***Examples***
D…	Discipline	
Di		Architectural
Dj		Structural
O…	Object	
Oi		Beam-1
Oj		Roof trusses
T…	Task	
Ti		Verify that pump size is adequate
Tj		Kitchen hoods and ducts are regulated by code
L…	Location	
Li		Living room
Lj		Master bedroom

Table 4-1 - Examples of designations given to the entities in each context

Using a context-aware rule, the logical links between the needed information, the 3D objects, the reviewers and their location in the environment, are established. Reviewer's comments and changes are captured and shared by others.

In describing the interactions between the search and retrieval of design review information, a design review of a 3-bedroom house from Shiratuddin & Thabet, 2007 is used, instead of the Virginia Tech's case study buildings. The author only had access to 2D drawings for the case study buildings. No 3D CAD models were created by the designers of the case study buildings. Inclusion of the case study buildings' 3D CAD models in the development of the WIP prototype design review application was not possible due to non availability and impracticality to create them.

Figure 4-11 shows an example of interactions between the search and retrieval contexts. Subsequent interactions are fission and finite in nature until the required information is found. In this example, the search starts with the discipline of interest to be reviewed. Dn where n can be i, j, k etc… simply denote the different disciplines designations (Table 4-1). Once a discipline is selected for review, information search and retrieval begins. The next level of interaction will either be between Dn and Tn, Ln or On.

Figure 4-11, Tn can either interact with Ln or On. A task in this case can be of an object On selected or the location Ln where the reviewer is.

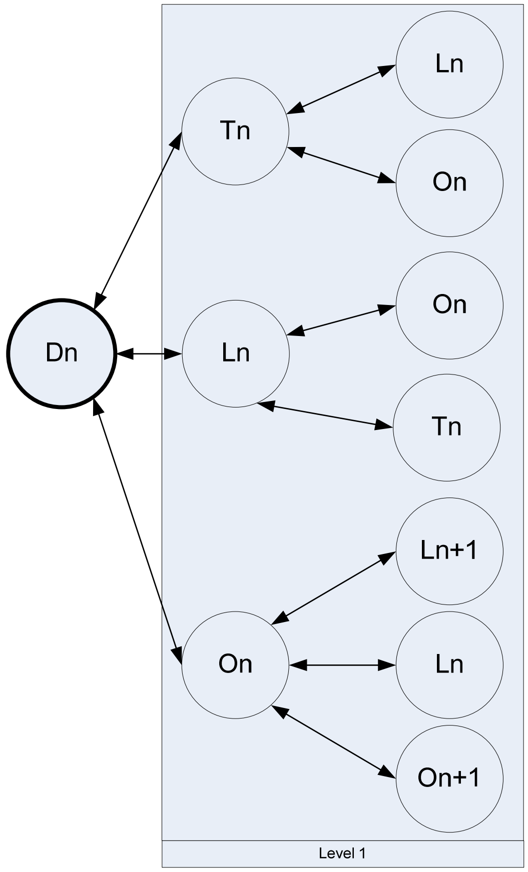


Figure 4-11 - An example of interactions between different contexts

Another form of interaction is between Dn and Ln. The location L centric is based on the location where the reviewer is. Once a location is detected and identified, information processing will again start. In Figure 4-11, the Ln centric can either interact with On or Tn. Once a reviewer is in the desired location, a reviewer can either retrieve information on object On or perform a task Tn that related to location Ln.

The next form of interaction is between Dn and On. In this particular example, On can then either interacts with object On+1 whereby On+1 is a neighboring/connected object, location Ln or location Ln+1. Location Ln+1 means that object On may be present in other location as well e.g. a ductwork that spans across multiple rooms, a duplex outlet installed in various locations, a bathroom fixture that is used in two bathrooms etc.

In summary, all possible interaction results in establishing a rule and this rule is used to retrieve the relevant information. Based on the interactions between the contexts, the rule will accommodate a more methodical approach in performing design review and is used for information processing. The rule uses the ***If, Then, And, Next, Else*** and ***Else If*** statements. Referring to Figure 4-12, an example of a rule would be:

If centric D = Di (Architectural)
    Then process information related to Architectural discipline

Next check centric interaction for T, L **And** O
    If centric = T
    And T = Ti
    Then load Architectural checklist
    And process information related Architectural checklist
    And process information on Li
    And process information on Oi

    Next check whether Li or Oi is selected
        If Li is selected
        Retrieve information on Li
        Display information on Li

        Else If Oi is selected
        Retrieve information on Oi
        Display information on Oi
Else check for L **And** O
    If centric = L
    And L = Li
    Then process information on Li
    And process information on Oi
    And process information on Oj (neighboring object)

Else check for O
    If centric = O
    And O = Oi
    Then process information on Oi
    And process information on Oj (neighboring object)
    And process information on Li
    And process information on Lj (Oi may also exist in Lj)

End

The design review application's rules are made up of many of smaller rules. From the rules produced, complex relationships and links among information will then be generated. These rules:

a) generate information relevant to a specific reviewer, related design review tasks, and assist in making decisions
b) help direct/navigate design review activities
c) recommend possible solutions based on the decisions reviewer have made in previous steps
d) inform reviewer if changes made will affect other adjacent or related components

Task-Centric Context Filtering

As abovementioned, a Task Centric uses criteria based on the objectives or tasks the reviewer would like to accomplish during design review.

Task Centric is used when a reviewer chooses to refer to a pre-defined design review checklist to perform the review. The checklist option is made available as a means to systematically guide the reviewers throughout the entire review process. The reviewer does not have to follow what is indicated in the checklist but rather use it as a reference.

Figure 4-12 shows main categories of information in the checklists relevant for the design review of the Structural Framing System for a 3-bedroom house.

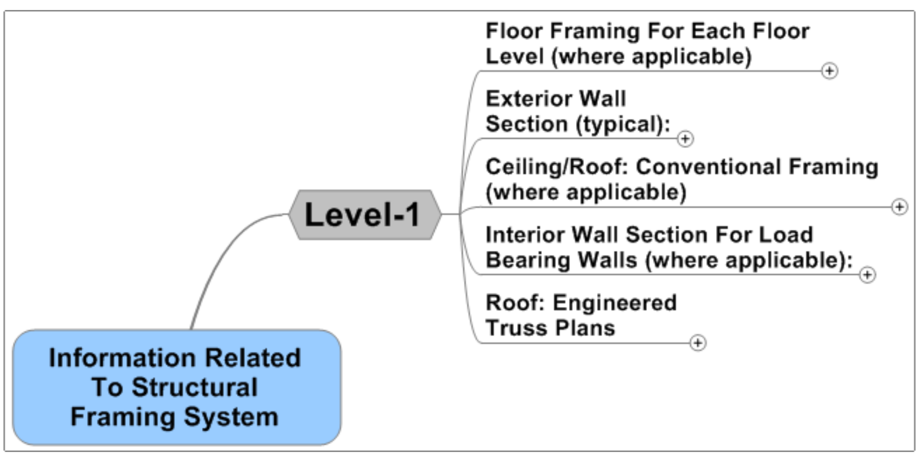


Figure 4-12 - Level-1 information of the Structural Framing System

Table 4-2 shows an example of a checklist for the exterior wall of the 3-bedroom house used in the WIP proof-of-concept prototype design review application: ***Exterior Wall Section → Type of wall → Masonry Wall***. Checklists categorized by discipline, assembly, location, and so on will be stored in the system. Items in the checklist can be either added or removed at anytime depending on the project's needs. Results from interviews show that design companies have their own customized checklist/s to review designs. To date, there is no known standardized checklist that is used by design reviewers.

		OK	n/a	Comments
1	whether vertical steel is used, if yes, the type of steel and the filling used in cells			
2	the type of window and lintels, the sizes and the reinforcement steel			
3	bond beam, size, reinforcement steel and treated top plate(if applicable) w/anchors			
4	identification of common wall/roof connector or see roof framing lay-out to clarify			
5	furring strips			
6	ceiling height			
7	4” min. required between grade and block			
8	fire rated walls (if) required and shown			

Table 4-2 - Example of checklist for Exterior Wall Section

The WIP prototype design review application allows the reviewer to turn on and off any objects in the VE such as shown in Figure 4-13. As such, the reviewer is able to see what is usually not visible due the nature of the construction assembly, e.g. items 1 to 8 in Table 2. In this example, by turning off the external wall layer (or the interior drywall layer), the reviewer will be able to perform a review on the specific items on the checklist.

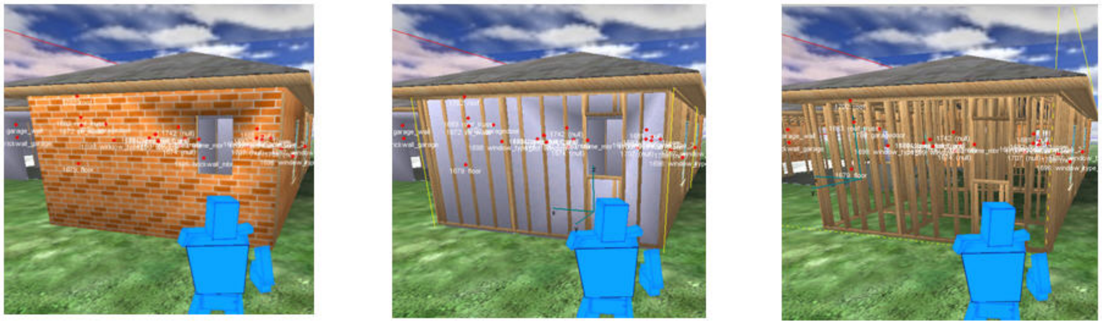


Figure 4-13 - A reviewer is able to turn on and off any object selected in the VE

Figure 4-14 shows the expansion of the main categories of information in the checklists relevant for the design review of the Structural Framing System for the 3-bedroom house. Each category of information can be expanded into three levels. For example Level-1 consists of the main categories of the framing system that are necessary to be reviewed by a structural reviewer:

1) the Floor Framing for each floor level
2) the Exterior Wall section
3) the Ceiling and Roof: Conventional Framing
4) the Interior Wall section for Load Bearing Walls, and

5) the Roof: Engineered Truss Plans

Depending on the applicability related to the design reviewer, and the detail of the design review process, Level-2 and Level-3 information in the checklists can be acquired.

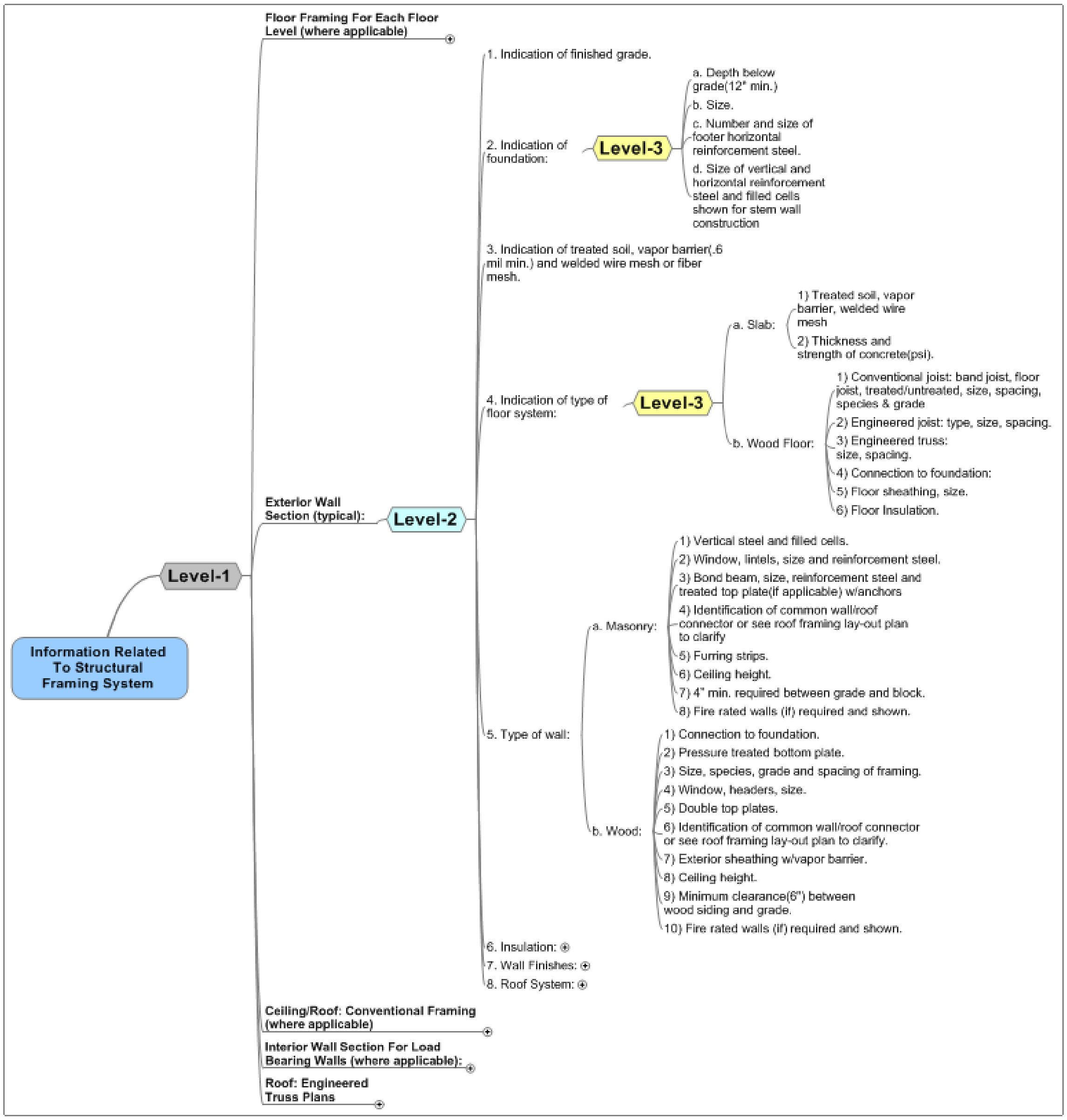


Figure 4-14 - Example of Level-1, Level-2 and Level-3 information in the checklists for the structural framing system of a 3-bedroom house

Figure 4-15 Figure 4-15 shows the possible interactions between the Task Tn centric with other centrics. Whenever a Task is the first action performed by a reviewer performed during design review, the Task Centric would be the primary node. The possible interactions between the Task Centric (acting as the primary node) and other nodes would be Tn to Dn, Tn to Ln or Tn to On. The second node which can be Dn, Ln or On can provoke further interactions with the third nodes. The interactions between the centrics are infinite until the required information is found, and this type of interaction is sometimes known as “fission reaction”. Figure 4-15 only shows Level-1 interactions that comprises of up to 3 nodes. Descriptive examples for the possible interactions are also shown in Figure 4-15.

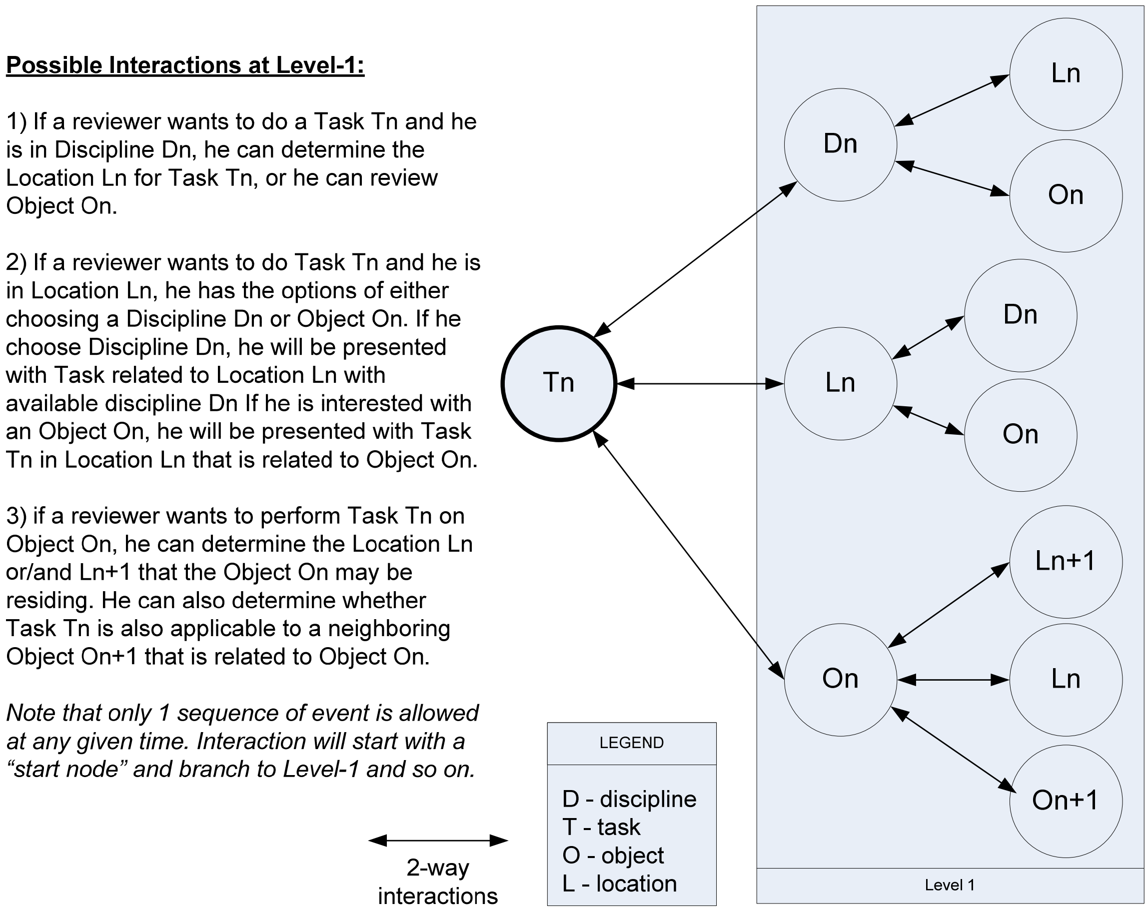


Figure 4-15 - An example of interactions between Task-Centric as the primary node with other centrics

Location-Centric Context Filtering

Location centric is based on the relative spatial position of the reviewer within the 3D model in the VE.

Location centric is used when a reviewer wants to get a general overview of a facility. The information filtered and presented will not be too detailed. For example, a structural system reviewer enters the VE with the intention to review the external facade of the 3-bedroom house. The relevant information he needs to visualize graphically and textually is the Level-2 information (Figure 4-15) i.e. the general indication of: 1) the finished grade, 2) the foundation, 3) any treated soil, vapor barrier (0.6 mil min.) and welded wire mesh or fiber mesh, 4) the type of floor system, 5) the type of wall, 6) the type of insulation, 7) the wall finishes, and 8) the roof system (since roof system is connected to the wall system).

A design review application should be designed in such a way that when a reviewer leaves a location and enter another, the application will present a cue alert. The location triggering mechanism (or markers) is placed at specific locations within an area of space, usually at the entrance such as a door or an opening. The triggering mechanism is important as it provides the data for the design review application to start filtering all the possible information required for a specific location in which the reviewer is currently located and wishes to review. Figure 4-16 shows an example of the placement of the location markers. Figure 4-17 shows how the markers are placed in the WIP proof-of-concept design review application.

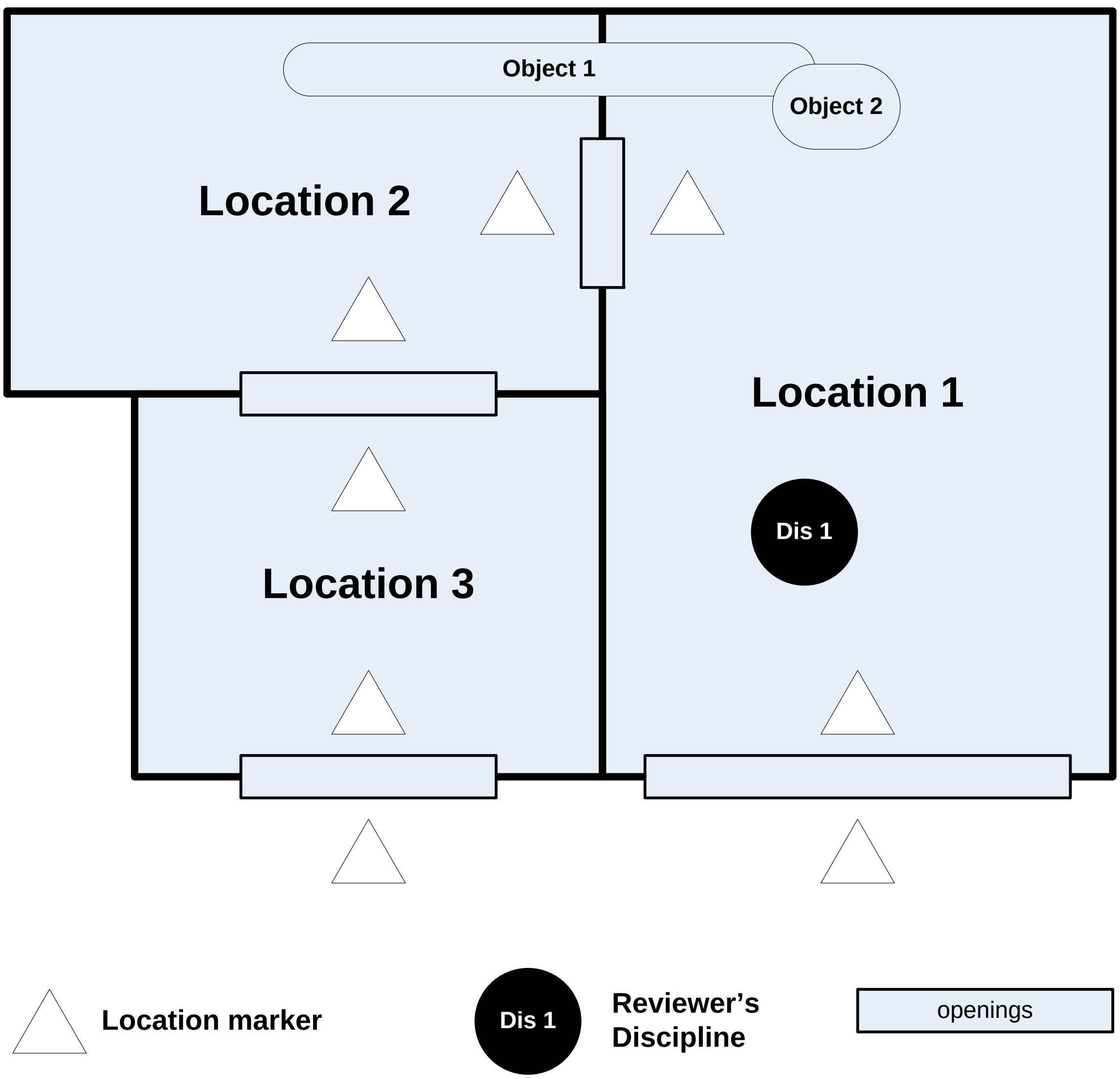


Figure 4-16 - Placement of the location markers in different locations

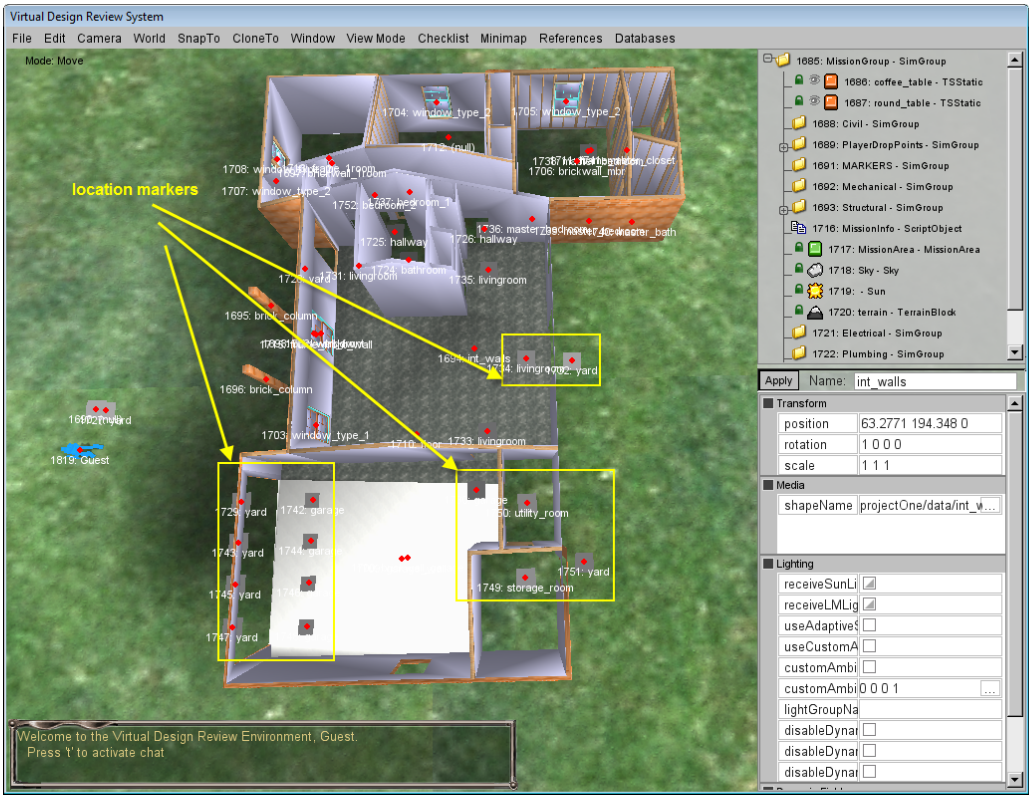


Figure 4-17 - Placement of the locations markers in the WIP proof-of-concept design review application

Figure 4-18 shows the possible interactions between the Location Ln centric with other centrics. Whenever a reviewer goes straight to a particular location Ln, the location Ln centric would be the primary node. The possible interactions between the Location Ln centric (acting as the primary node) and other nodes would either be Ln to Tn or Ln to On.

The second node which can be Dn, On or Tn can instigate further interactions with the third nodes. The interactions between the centrics are infinite until the required information is found. Figure 4-18 only shows a Level-1 interaction that comprises up to 3 nodes. Descriptive examples for the possible interactions are also shown in Figure 4-18.

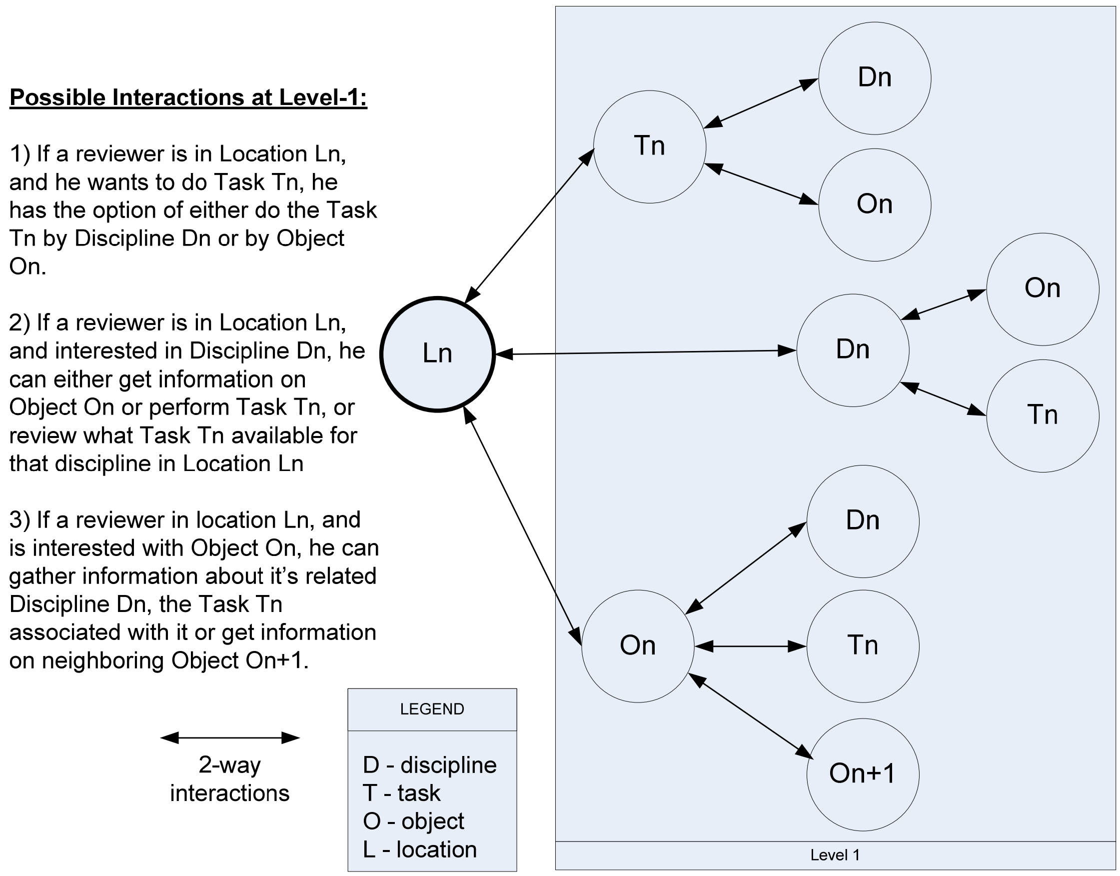


Figure 4-18 - An example of interactions between Location-Centric as the primary node with other centrics

Object-Centric Context Filtering

The object centric uses criteria based on the graphical component/assembly selected in the VE by the reviewer for review.

Object centric is used when a reviewer decides to review a specific object in the VE. For example, a structural reviewer wants to know the details of the rafter used for the virtual 3-bedroom house. The reviewer can directly click on the ceiling/roof object. A design review application should return the information on locations of the rafters, ceiling joist, ridge beam, and collar tie, size, species, grade and spacing in the form of pop-up window (Figure 4-19). Figure 4-19 also shows a how a design reviewer interacts with the information of an object in the VE.

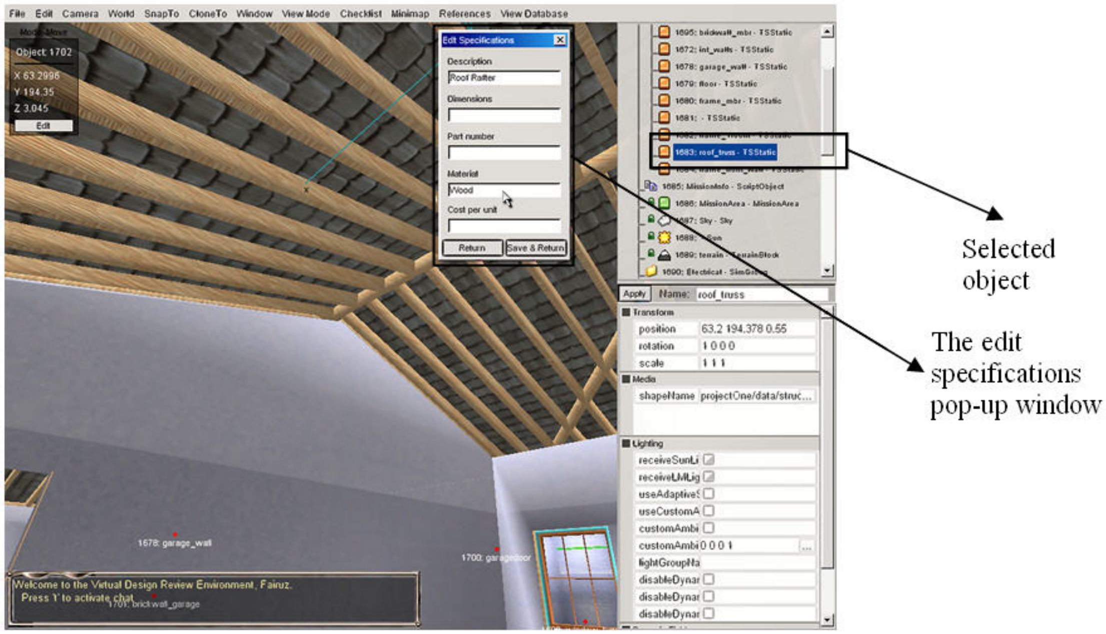


Figure 4-19 - Reviewer is able to select individual object in the VE and review its specifications

Figure 4-20 shows the possible interactions between the Object On centric with other centrics. Whenever a reviewer selected an Object On and it is the first action performed, the Object On would be the primary node. The possible interactions between the Object On centric (acting as the primary node) and other nodes would be On to Dn, On to Ln or On to Tn. The second node which can be Dn, Ln or On can instigate further interactions with the third nodes. Again, the interactions between the centrics are infinite until the required information is found. Figure 4-20 only shows a Level-1 interaction that comprises up to 3 nodes.

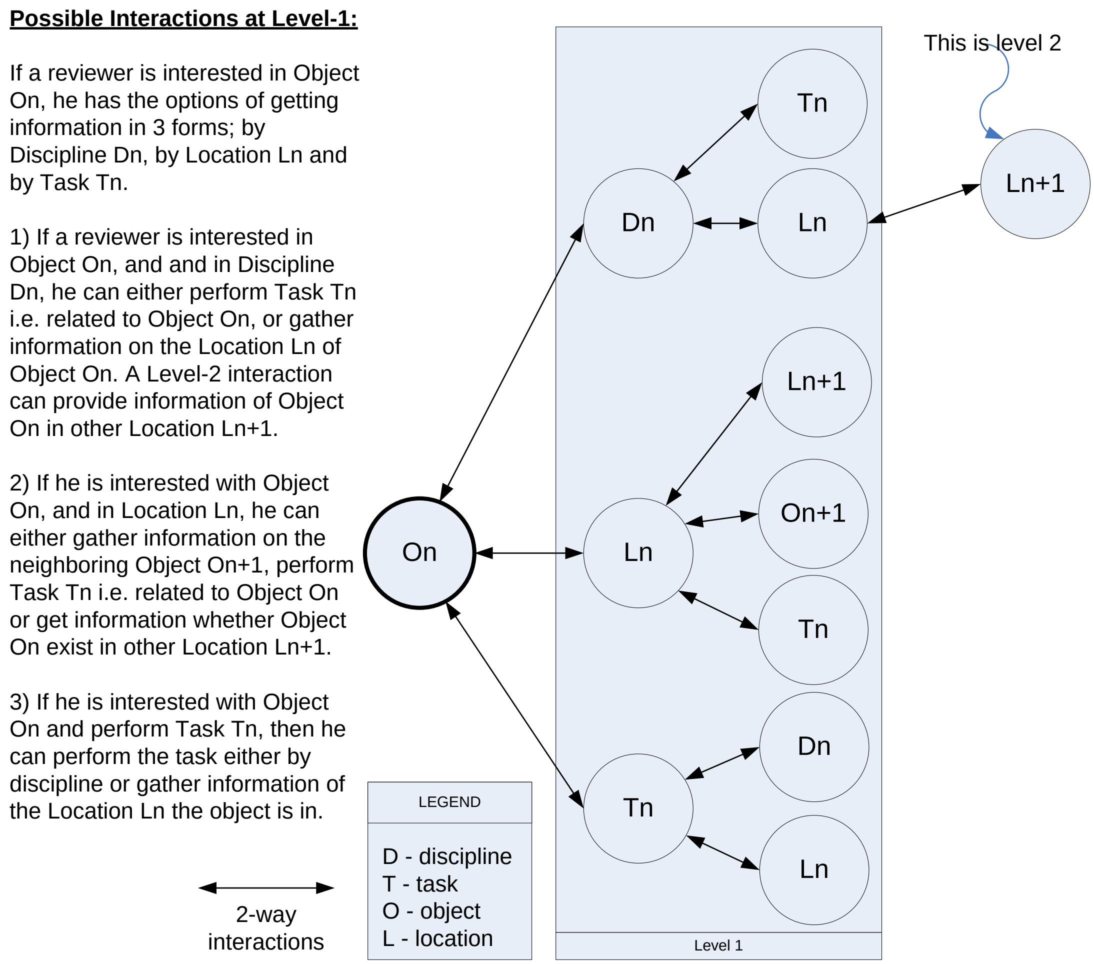


Figure 4-20 - An example of interactions between Object-Centric as the primary node with other centrics

Discipline-Centric Context Filtering

Figure 4-21 shows the implementation of the login screen of the WIP proof-of-concept design review application. The login screen allows a reviewer to define a role and discipline of review interest. Once logged in, a design review application should present the required view of the 3D model in the VE. For example, a structural engineer is interested in reviewing the structural system of the 3-bedroom house hence the design review application will initially present only the structural system. However, should the structural reviewer need to review other building systems, the application should allow for other 3D objects from other building systems to be displayed along with the structural system.

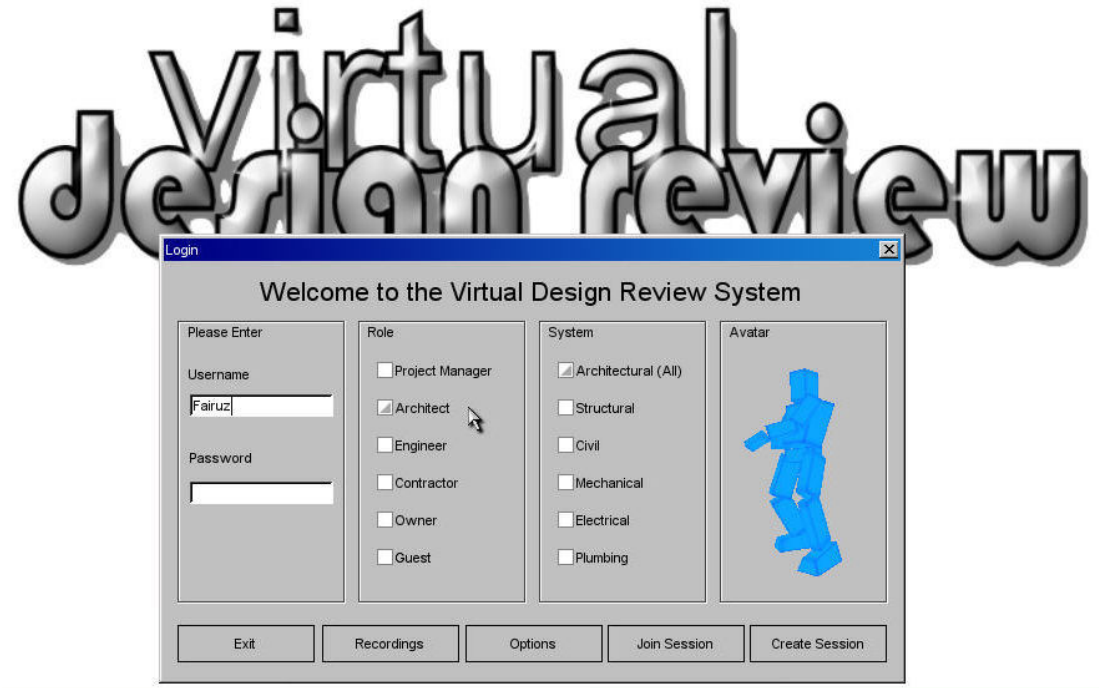


Figure 4-21 - Login screen to cater for discipline-centric information processing

Figure 4-22 shows the possible interactions between the Discipline Dn centric with other centrics. Whenever a reviewer chooses to review for Discipline Dn, then Discipline Dn would be the primary node. The possible interactions between the Discipline Dn centric (acting as the primary node) and other nodes would be Dn to Tn, Dn to Ln or Dn to On. The second node which can be Tn, Ln or On can instigate further interactions with the third nodes. Figure 4-22 only shows a Level-1 interaction that comprises up to 3 nodes.

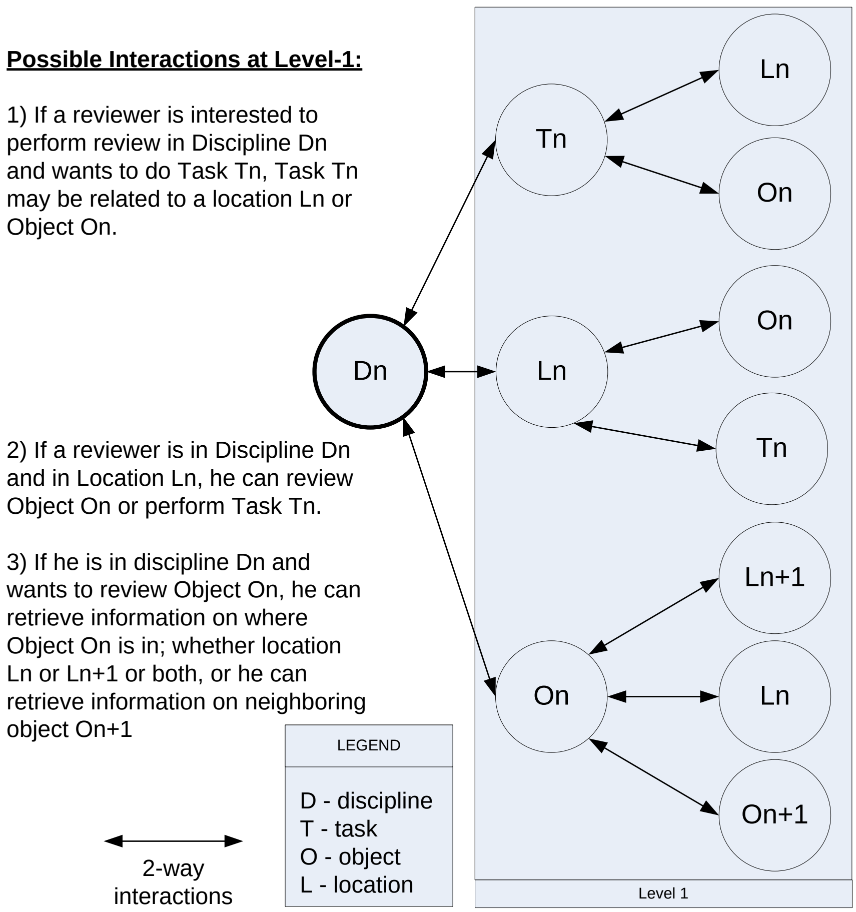


Figure 4-22 - An example of interactions between Discipline-Centric as the primary node with other centrics

a) Illustration of Design Review Application in a VE based on Case Study

As mentioned in Chapter 3, there is no consistent pattern as far the occurrences of errors in designs are concerned. Based on this fact, this sub-section outlines selected errors taken from the three Virginia Tech's past project. The examples depict the various interactions among the components of the design review framework and the design reviewer. The examples are to further clarify the inner-working of each context filtering mechanism.

Example 1: Electrical Systems Design (Task Centric with Object as Sub-Centric)

A summary on how the design review system responds to design reviewer interaction that requires a task-centric context filtering is shown below.

This is an example of an Electrical Design reviewer who reviews Electrical Design of a facility, specifically on the fire protection issues. The reviewer found an error in design i.e. a missing exit sign on exit doors.

Process	**Action**
Reviewer login	>Input ID, Discipline (Electrical), Project Name, etc.
System Response	>Load 3D model of facility in VE >Presents Walkthrough mode
Reviewer switch	>Reviewer switch to Walkthrough in review mode (F11)
Reviewer select	>Select "Electrical system design checklist"
System Response	>Query Database Checklist >Load and Display window for "Electrical system design checklist"
Reviewer review	>Go through line item #1a in checklist: • View lighting layout for each floor and check for: Fixture Location (lamp, fan, exit sign) >Select option to review lighting layout
System Response	>Display lighting layout >Hide the rest of building systems
Reviewer review	>Highlight 3D Lighting layout/3D electrical objects >Review 3D objects >Found Error (i.e. missing exit sign on a door) >Reviewer queried (keyword entry) information on "Exit Sign" from system

System Response	>Query database >Display results to reviewer with all the relevant information e.g. codes, project specs etc.
Reviewer review	>Confirm codes and pick the most relevant code >Input comments in comment form: "Provide an exit sign for the door to the exit stair and illumination per Sections 604.2.3, 604.2.4,1011.5.2 and 1011.5.3 1.1 International Fire Code 2003"
System Response	>Store comments in Comments Database >End for line item #1
System Response	>System show checklist, move on next line item
Reviewer review	>Review next line item in checklist, or review other object

**References retrieved are then displayed to reviewer. Please refer to Appendix 2 for more details on the codes used in this example.*

The above description is shown in the flow-diagram in Figure 4-23.

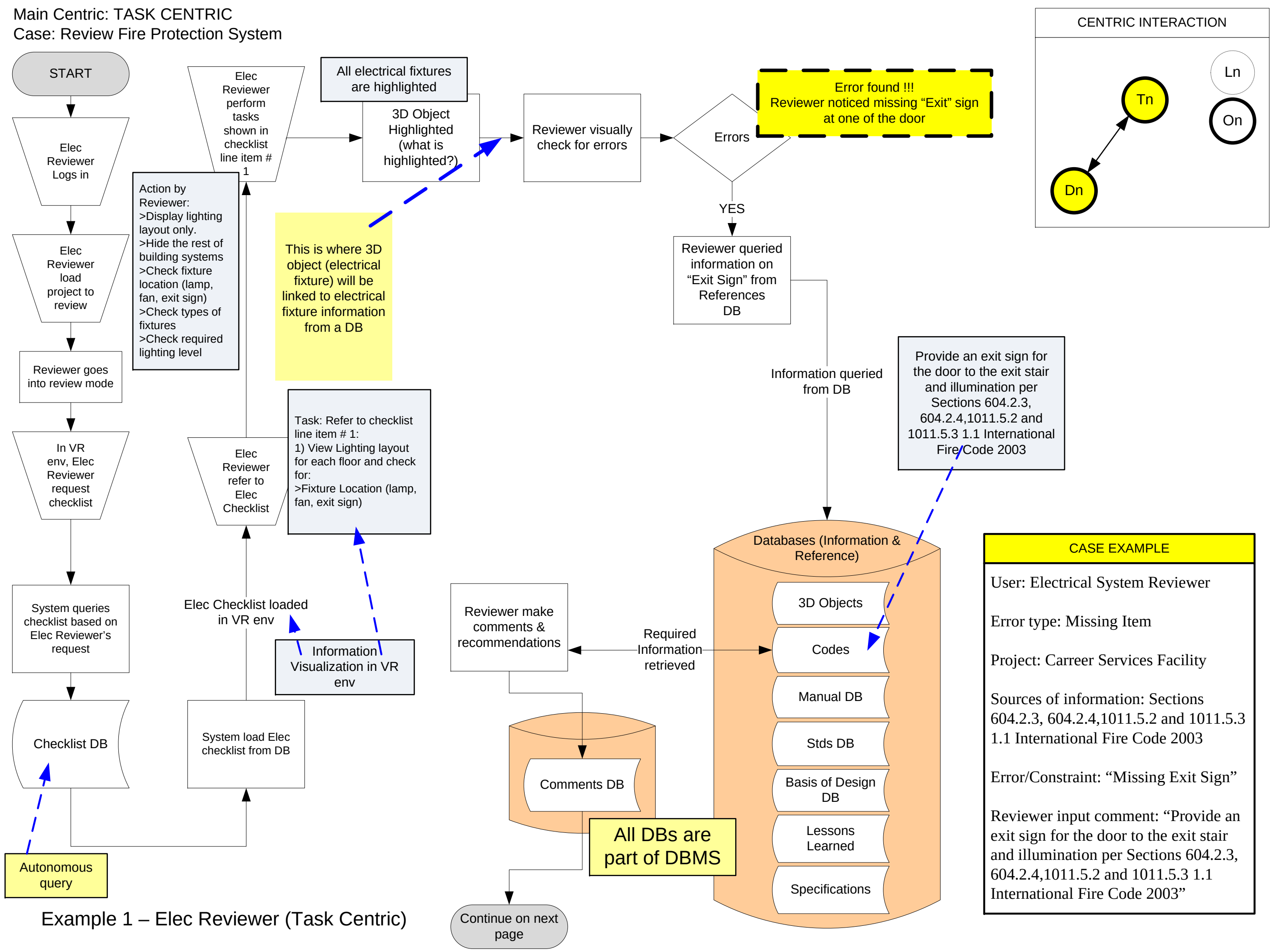


Figure 4-23 - Flow-diagram for Example 1

Example 2: Electrical Systems Design (Task Centric with Object as Sub-Centric)

A summary is shown below on how the design review system responds to design reviewer interaction that calls for transition from task-centric to object-centric context filtering.

This example is a continuation from Example 1. It shows the reviewer has completed the first line item in the checklist (where the system is responding in a task-centric context filtering) where he found there was an error. The reviewer then switches to an action that requires the system to respond in object-centric context of filtering information.

Reviewer's last action was to input his comments in the comment form, and continue to the next line item in the checklist. He then decides to review particular object/fixture fluorescent lamp that is in the line item #2.

Process	**Action**
Reviewer review	>Review next line item #2 in checklist >Selects object/electrical fixture fluorescent lamp
System Response	>Link 3D object (selected electrical fixture fluorescent lamp) with information of electrical fixture in Specifications database >Display attributes of fixture: • Lumens: • Location: • Wattage per hour: • Light emission color: • Part #:
Reviewer review	>Check specification >Found no error >End >(Choices) Either go back to Electrical System Design Checklist i.e. review next line item in checklist, or review other object or End Session
System Response	>System show checklist, move on next line item or End Session

**References retrieved are then displayed to reviewer. Please refer to Appendix 2 for more details on the codes used in this example.*

The above description is shown in the flow-diagram in Figure 4-24.

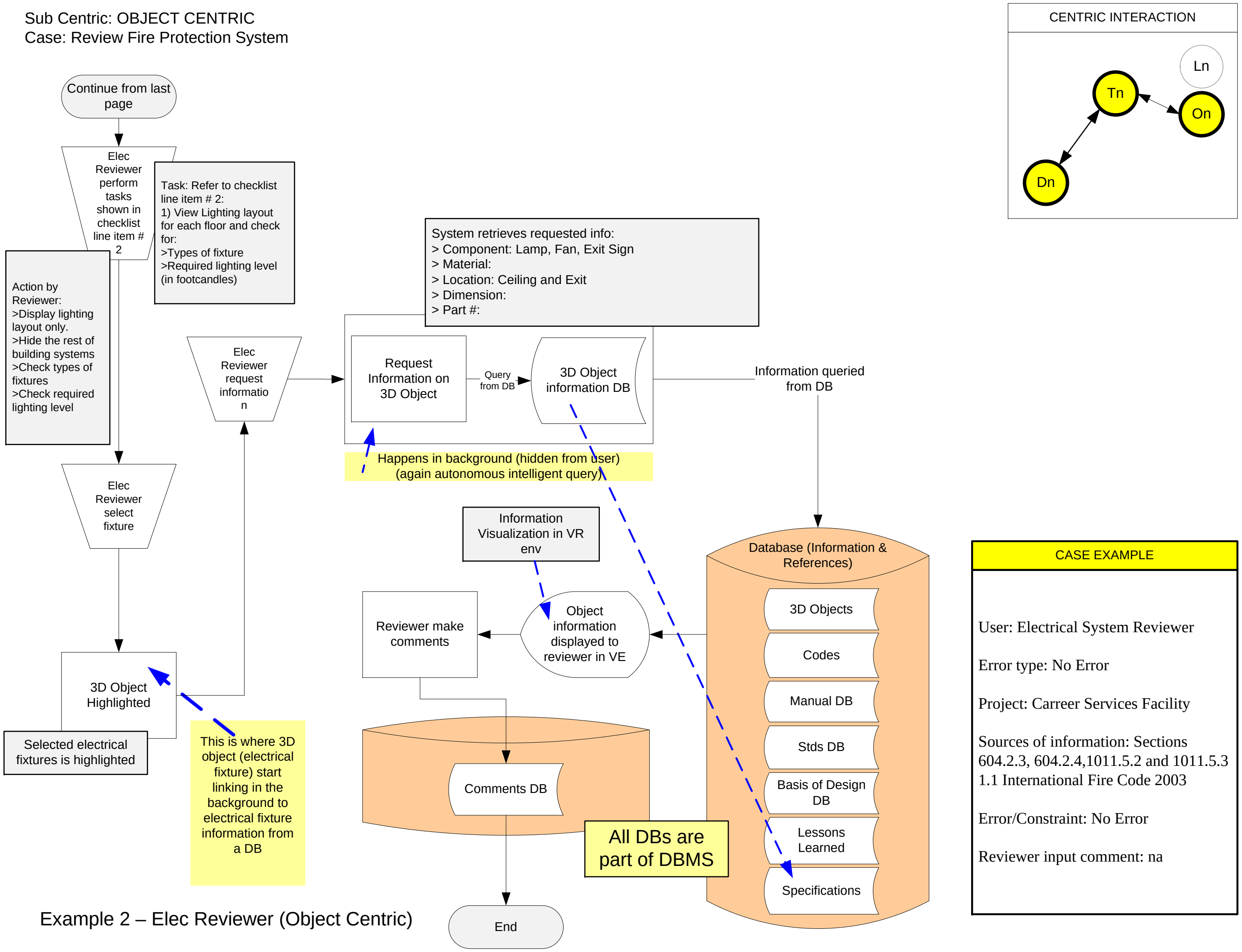


Figure 4-24 - Flow-diagram for Example 2

Example 3: Roof Live Snow Load (Object Centric)

A summary is shown below on how the system responds to design reviewer interaction that requires an object-centric context filtering. The reviewer is a Structural Design reviewer who comes in and reviews the roof construction of the facility. The reviewer found an error in the roof construction.

Process	**Action**
Reviewer login	>Input ID, Discipline (Structural), Project Name, etc.
System Response	>Load 3D model of facility in VE >Presents Walkthrough mode
Reviewer switch	>Reviewer switch to Walkthrough in review mode (F11)
Reviewer select	>Select Roof

System Response	>Highlight roof object/component
	>Link 3D roof object with information in Specifications database.
	>Display roof attributes:
	• Material:
	• Location:
	• Dimension:
	• Specs:
	    • Live loads:
	    • Wind loads:
	    • Seismic criteria:
	• Part #:

Reviewer review	>Review roof >Decide to review via roof portion of Checklist >Select Structural Checklist from main menu

System Response	>Retrieve and display roofing portion of Structural Design
	>Checklist
	>Display line item #1 of checklist:
	• Are the following items designed per VUSBC Chapter 16?
	    • live loads (y/n/na/comments/recommendation)
	    • wind loads (y/n/na/comments/recommendation)
	    • seismic criteria (y/n/na/comments/recommendation)

Reviewer review	>Review Structural Checklist line item #1: >Review roof attributes >Found error >Reviewer queried (keyword entry) on Live Load codes from system
System Response	>Retrieve information on Live Load codes >Display results to reviewer (in form of sections of codes) *
Reviewer review	>Confirm codes and pick the most relevant one >Input comments in comment form: “Roof live load is 20 psf, >should be changed to 30 psf per CPSM 701.9.8”
System Response	>Store comments in Comments Database >End >System show checklist of next line item
Reviewer review	>Review next line item in checklist, or review other object or End Session

**References retrieved are then displayed to reviewer. Please refer to Appendix 2 for more details on the codes used in this example.*

The above description is shown in the flow-diagram in Figure 4-25.

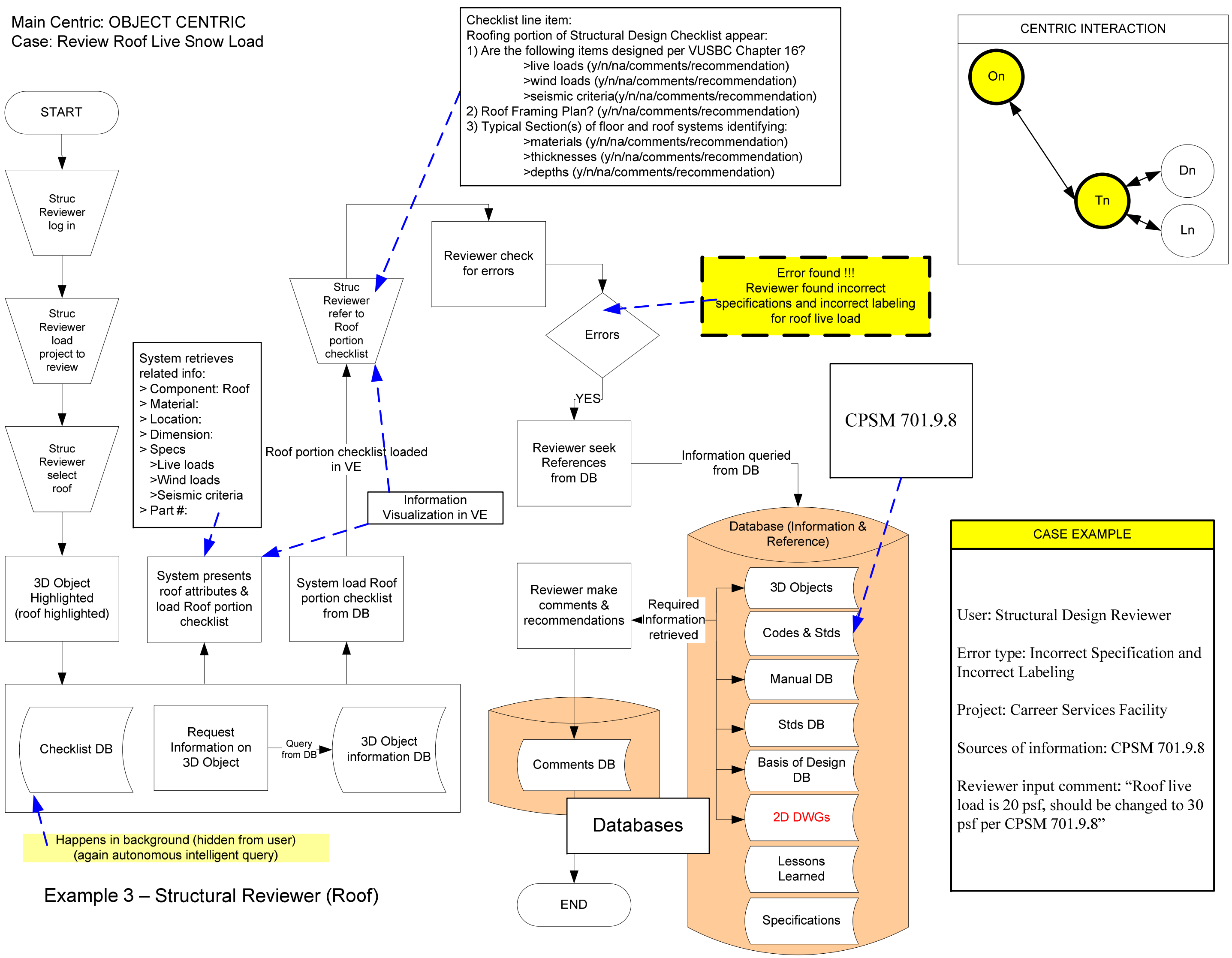


Figure 4-25 - Flow-diagram for Example 3

Example 4: Code Compliance of Restroom Fixture (Location Centric with Object as Sub Centric)

Summary is shown below on how the system responds to design reviewer interaction that calls for a location-centric context filtering. The reviewer is the Architectural Design reviewer and specifically reviews the restrooms for the facility.

Process	**Action**
Reviewer login	>Input ID, Discipline (Architectural), Project Name, etc.
System Response	>Load 3D model of facility in VE >Presents Walkthrough mode
Reviewer switch	>Reviewer switch to Walkthrough in review mode (F11)
Reviewer select	>Walkthrough to restroom
Reviewer review	>Select toilet tissue dispenser >Query toilet tissue dispenser specs
System Response	>System highlight toilet tissue dispenser
System Response	>Query Specifications database >Display toilet tissue dispenser attributes:
Reviewer review	>Found Error >Query code database
System Response	>Query database >Display UFAS code >Display Lessons Learned: >“This same problem came up on another agency's project in 1997. On November 25, 1997 BCOM received a drawing from the Department of Justice (Ellen Harland, AIA) showing the toilet tissue dispenser could not project more than 6" into the handicapped toilet stall space. VPI & SU can use the 13" roll as their standard in all the non-handicapped toilet stalls only”.
Reviewer review	>Confirm codes and pick the most relevant one >Input comments in comment form: “This 13"x14" toilet tissue dispenser is not to be used in the handicapped toilet stalls. Use the standard small roll toilet tissue dispenser. See Figure 30 (d) in the Uniform Federal Accessibility Standards. UFAS section 4.16.6 requires a continuous paper flow and uses Figure 29 (b) to show the smaller toilet tissue dispenser. Please make correction”.

System Response	>Store comments in Comments Database >End >System shows checklist of next line item or End Session.
Reviewer review	>Review next line item in checklist, or review other object or End Session.

**References retrieved are then displayed to reviewer. Please refer to Appendix 2 for more details on the codes utilized in this example.*

The above description is shown in the flow-diagram in Figure 4-26.

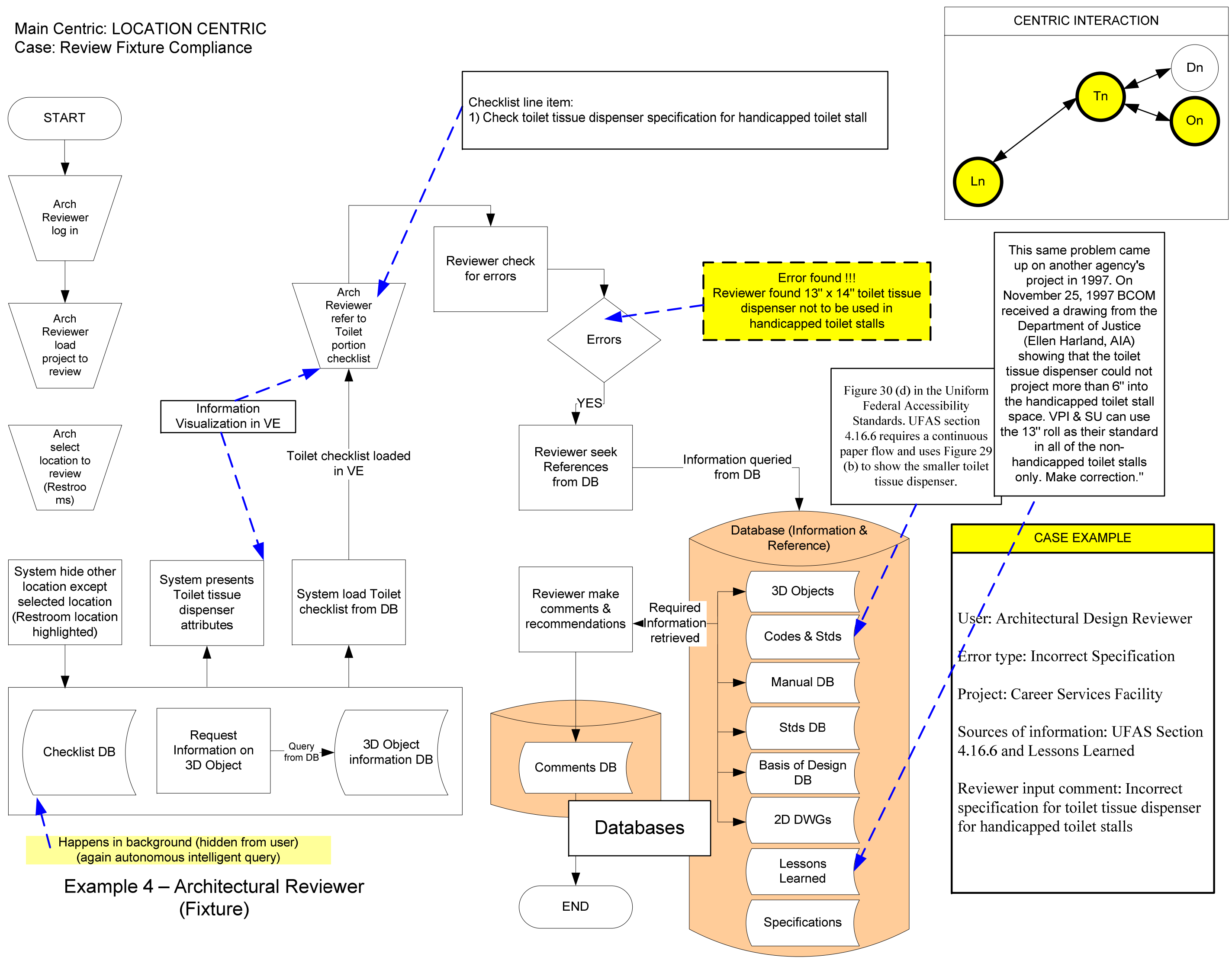


Figure 4-26 - Flow-diagram for Example 4

Example 5: Relocate items (Location Centric)

A summary is shown below on how the system responds to design reviewer interaction that calls for a discipline-centric context filtering. The reviewer is the User Representative and specifically reviews the entire facility.

Process	Action
Reviewer login	>Input ID, Discipline (User Representative), Project Name, etc.
System Response	>Load 3D model of facility in VE >Presents Walkthrough mode
Reviewer switch	>Reviewer switch to Walkthrough in review mode (F11)
Reviewer select	>Walkthrough to restroom
Reviewer review	>Select "Switch and Junction box"
System Response	>System highlight "Switch and Junction box"
Reviewer review	>Query "Switch and Junction box" from project specs database
System Response	>Query project specs database >Display "Switch and Junction box" attributes: • Location: • Quantity/room: • Part #:
Reviewer review	>Found error in the location of switch and junction box >Input comments in comment form: >"The screen should be located on the west wall and the switch and junction box near the southwest corner"
System Response	>Store comments in Comments Database >End >System show checklist, or next line item or End Session
Reviewer review	>Review next line item in checklist, or review other object or End Session

**References retrieved are then displayed to reviewer. Please refer to Appendix 2 for more details on the codes utilized in this example.*

The above description is shown in the flow-diagram in Figure 4-27.

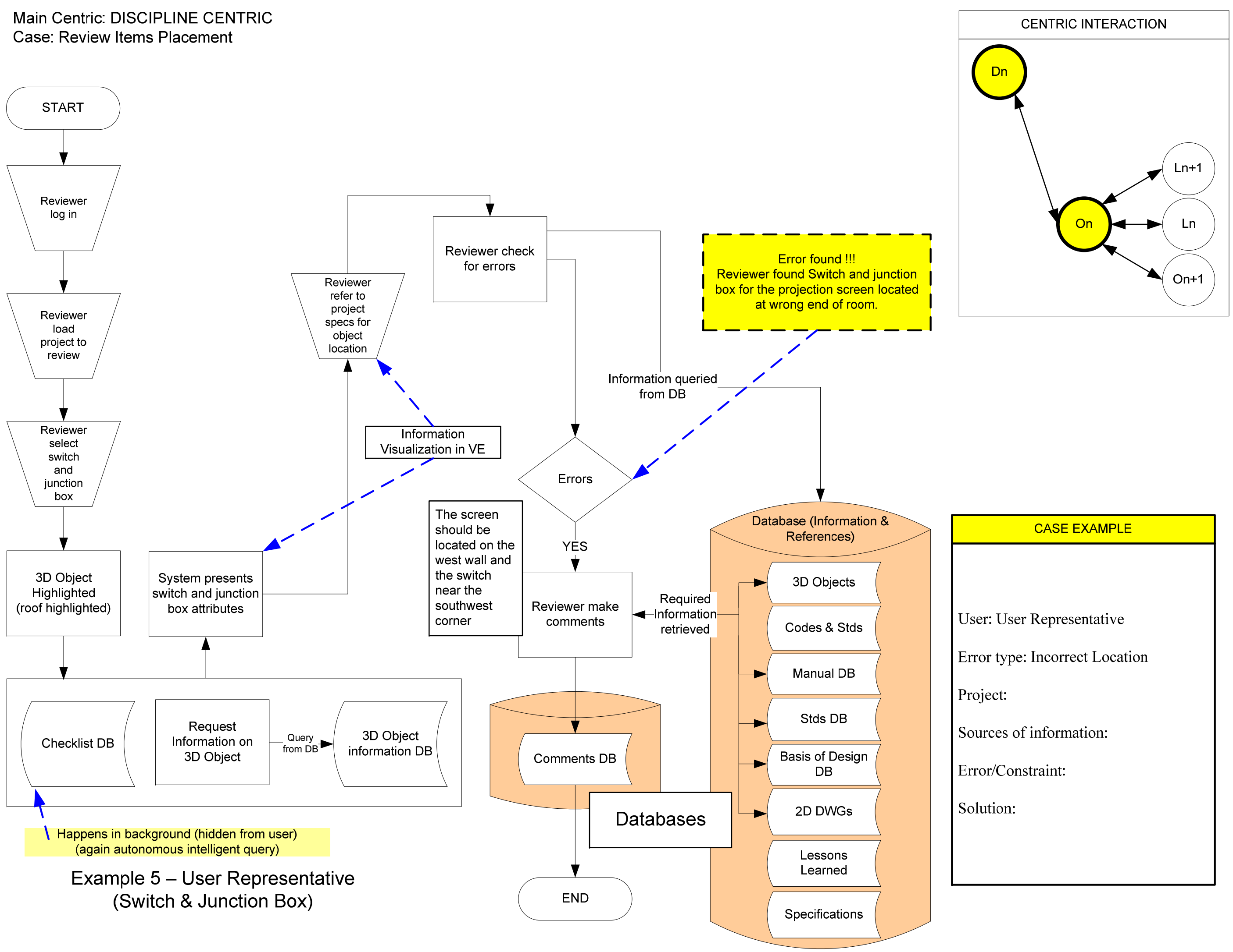


Figure 4-27 - Flow-diagram for Example 5

Example 6: Error in Plumbing Material (Task Centric)

A summary is shown below on how the system responds to design reviewer interaction that calls for a task-centric context filtering. This is an example of a Plumbing Design reviewer who reviews the plumbing material of the facility. The reviewer found an error in the material of pipes.

Process	**Action**
Reviewer Login	Input ID, Discipline (Plumbing), Project Name, etc.
System Response	Load 3D model of facility in VE Presents Walkthrough mode
Reviewer switch	Reviewer switch to Walkthrough in review mode (F11)
Reviewer select	Walkthrough to restroom
Reviewer review	Review skeletal layout of plumbing design Select floor slab
System Response	Retrieve attribute for floor slab Highlight floor slab Display attributes: • Component: Pipe • Material: PVC • Location: under floor slabs • Dimension: • Part #:
Reviewer review	Review floor slabs Found Error that under floor pipe is made of PVC Query Plumbing Codes
System Response	Query Plumbing Codes Display results of Plumbing Codes to reviewer (in form of sections of codes)*: • Uniform Plumbing Code (UPC) [Ch 3] • Sanitary Piping must be cast iron to at least 5'-0" beyond building wall
Reviewer review	Confirm Error Input comment in review form “ Should use cast iron for pipes under floor slab”
System Response	Store comments in Comments Database

	End System show checklist, or next line item or End Session
Reviewer review	Review next line item in checklist, or review other object or End Session

**References retrieved are then displayed to reviewer. Please refer to Appendix 2 for more details on the codes utilized in this example.*

The above description is shown in the flow-diagram in Figure 4-28.

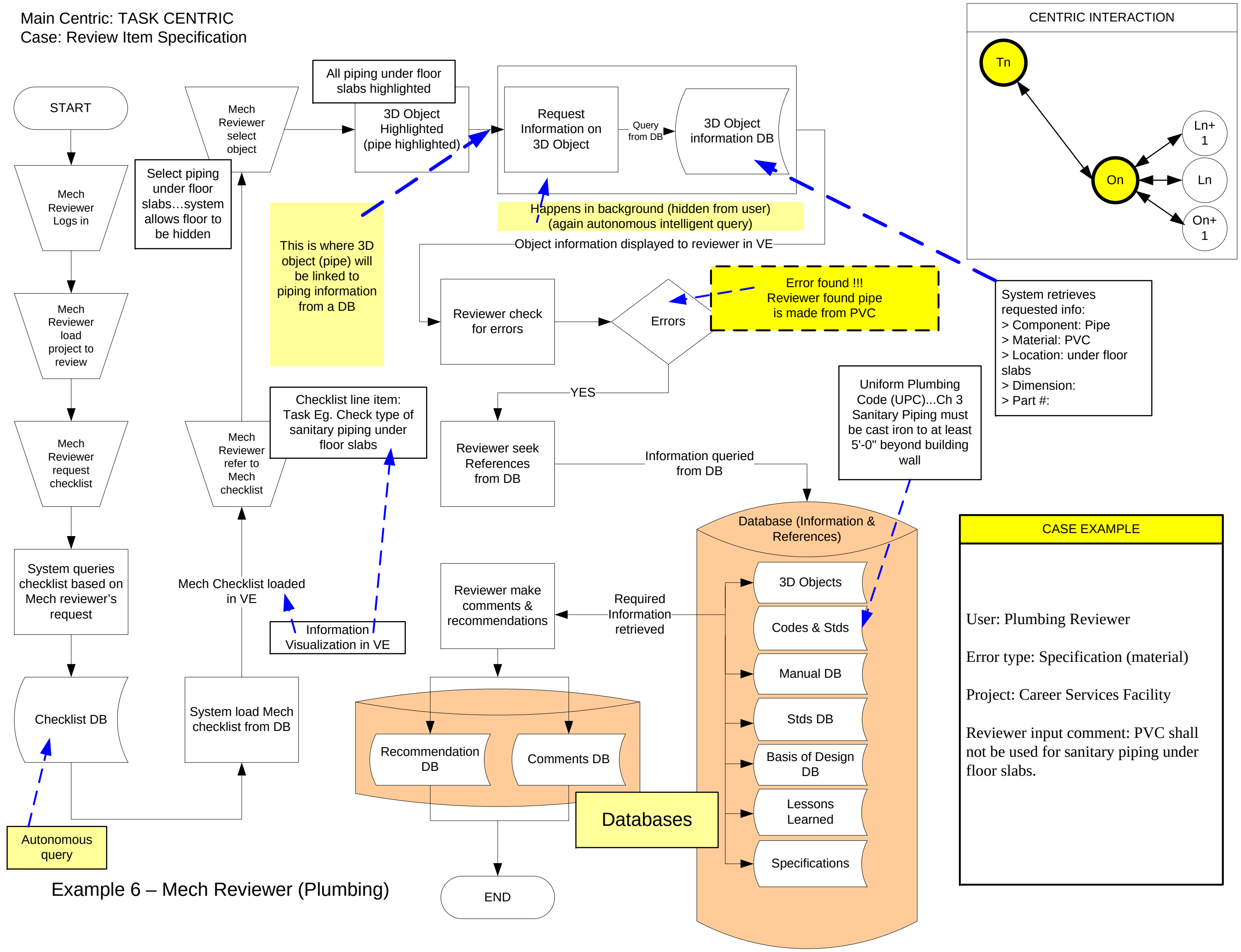


Figure 4-28 - Flow-diagram for Example 6

4.2.2 Virtual Environment

Shown in Figure 4-1 are the components of the VE portion of the framework. It consists of 3D Objects or CAD model, virtual checklists, and information in form of textual, abstract, and geometric. Abstract information refers to information which is invisible from the view of design reviewers. The abstract information is in the form of “raw data” which on its own maybe meaningless and unusable for design reviewers, e.g. a ‘comment_ID’, unless it is processed and tied to a task, location, discipline or object during design review. Once the abstract information is processed in the background, it is then visually presented in the VE in the form readable textual information.

4.2.2.1 GUI and Visualization of Design Review Information in the Virtual Environment

The author addresses Galitz (1997) and Mandel (1997) emphasis on the importance for well designed information display. Displayed information is the visible components that the reviewer a user connects/relates to. The author follows Mandel (1997) Golden Rules of Interface Design in the new design review approach

- Design reviewers are given the full control of the prototype’s user interface. In the design review mode, the main menu is designed to be visible at all-time. Meaningful commands are placed in the main menu e.g. File, View, Checklist, References etc. For the ease of navigating in the VE a keyboard and mouse combination is used. Using the W, S, A, D keys and the mouse, users are free to go where they want to go in the VE.
- Design reviewer’s memory load is reduced. This is when the computer can assist the user to remember and carry out tasks that are repetitive. E.g. defaults such as, undo, redo, copy, and the use of visible and meaningful icons.
- Consistency in layout of the user interface. Consistent layout leads to usable user interface where design reviewer can reuse their knowledge from other familiar software applications. E.g. the prototype application implemented by the author e.g. uses drop-down menus and right-click-popup menus. Many AutoCAD users are familiar with such concept.

The author ensures the GUI design provide design reviewer with the tools and applications to do a task. Design reviewer can directly manipulate objects and information to do the task on-screen. The author addresses Helander (1988), Mandel (1997), Galitz (1997), and Benbasat & Todd (1993)’s emphasis on the importance of properly presenting information to design reviewers. The author ensures that:

- The GUI displays information in an uncrowded manner. Information which is displayed in less crowded manner improved information clarity and readability. The prototype also displays only the information the design reviewer needs in order to perform his/her task or operation. Unnecessary details result to distraction and more amount of time needed to complete tasks.

- The information is displayed in proper groupings. The GUI of the prototype uses simple window, pull-down menu, pop-up window and icons. This is achieved by using small icons or windows that do not consume much space. Design reviewers can easily recognize the relationships of information.
- Information is displayed in logical sequencing. The layout of information on display is arranged in a way that is easy for the design reviewer to navigate and find the command and information he/she needs to perform design review tasks. E.g. the use of pull-down menu from the Main Menu bar allows design reviewer to perform various tasks such as saving a session, system settings, checklist menu, access to references etc.

4.2.2.2 Illustration of Reviewer Interaction using the WIP Prototype Application

Below is an example of the steps a design reviewer take when using the prototype design review application and the processes that happen in the background within the system. A desktop-PC is used in this example.

1) Prior to using the prototype design review application, the system administrator (usually the Project Manager) gives authorization and provides the reviewer with a username and password.

2) The reviewer logs into the WIP prototype design review application by keying a username and password, select the role and the building system he wishes to review (Figure 4-29). The WIP prototype design review application then authenticates and verifies the username and password, and allows the reviewer to use the system. In this example, the Role of Project Manager is selected, and Discipline to review is Architecture.

   Intelligent information filtering starts here with Discipline → Architecture being the context.

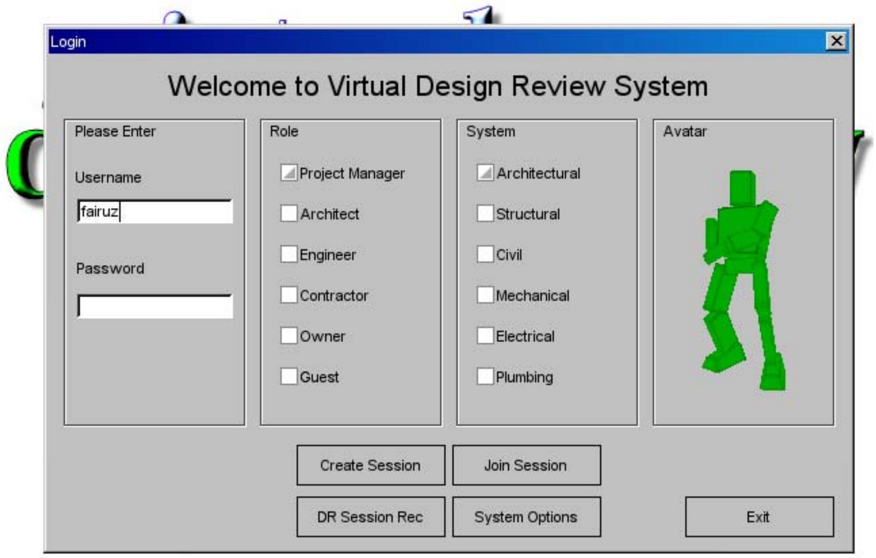
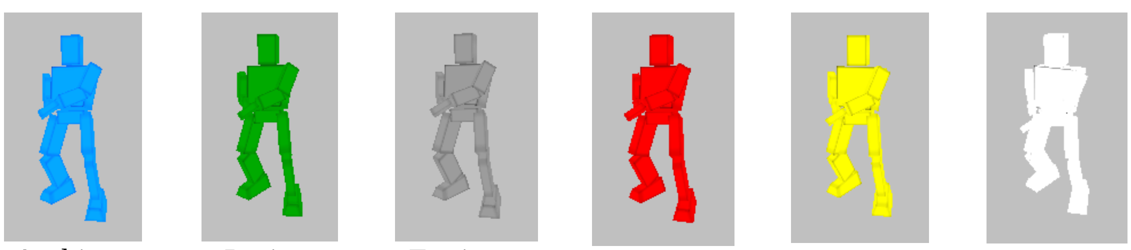


Figure 4-29 - The WIP prototype design review application login screen

3) In the background, the WIP prototype design review application logs the date and time when the reviewer starts using the system. It assigns an avatar with specific avatar for specific role e.g. a green-human-figure represent a Project Manager. The application also accesses the information of the reviewer and attaches it to the avatar. It then gathers and prioritizes the required information for the reviewer to perform design review. When multiple reviewers use the WIP prototype design review application at the same time in a collaborative fashion (local and remote reviewers connected across the network in real-time), the different disciplines of each reviewer is identified with different avatar colors (Figure 4-30).



Architect (blue)	Project Manager (green)	Engineer (grey)	Contractor (red)	Owner (yellow)	Guest (white)

Figure 4-30 - Different avatar colors to depict the role of the reviewers

4) The load project screen is then presented to the reviewer to allow him to select the desired project for review (Figure 4-31). Using the mouse as the interaction device, the reviewer clicks (left-mouse click) on the 'Single Family Home – 3 Bedroom, 2 Bathroom' project from the list of project available. Since the WIP prototype design review application allows for different iterations to be saved based on different sessions, information based on date, time, and personnel who made the changes are stored in the database.

If a reviewer wishes to review a project with another remote reviewer i.e. in a collaborative design review fashion, he can check on the ‘Host Design Review Session’ option.

Once satisfied with all the options available in the load project window, the reviewer then clicks on the ‘Launch Session’ button.

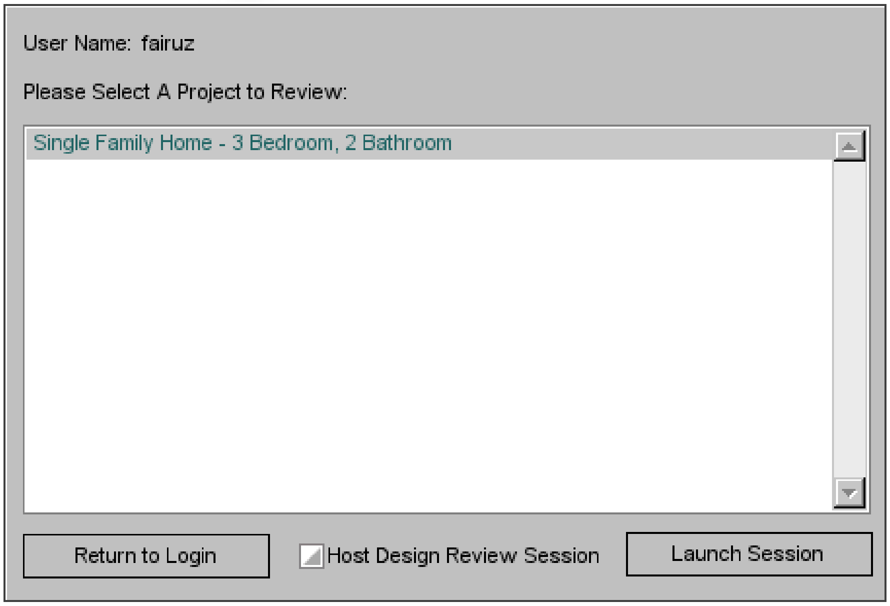


Figure 4-31 - The load project screen

5) The project’s 3D model is then loaded in the VE. By default, the application always launches in the “Walkthrough” viewing mode (Figure 4-32). This viewing mode allows the reviewer to be familiar with the project since no changes are allowed in this mode. A reviewer can explore the 3D model in VE, either by walking through, flying through, and use different predefined camera mode such as bird’s-eye view, first-person view, third-person view, orbital view or behind-the-shoulder view of another reviewer.

Figure 4-32 - The project's 3D model when first loaded. The "Walkthrough" mode is the default viewing mode

6) To start performing a design review task, the reviewer goes into the "Review" mode by pressing the F11 key. In the review mode, various design review tasks can be performed. The tasks include placing comments, loading and going through a design review checklist, object manipulation, collaborate with other reviewers across the network in real-time, and access the required design review related information.

7) In this example the reviewer loads the Architectural design review checklist from the drop-down main menu bar. The review then clicks on `Checklist' and selects a 'Architectural' (Figure 4-33).

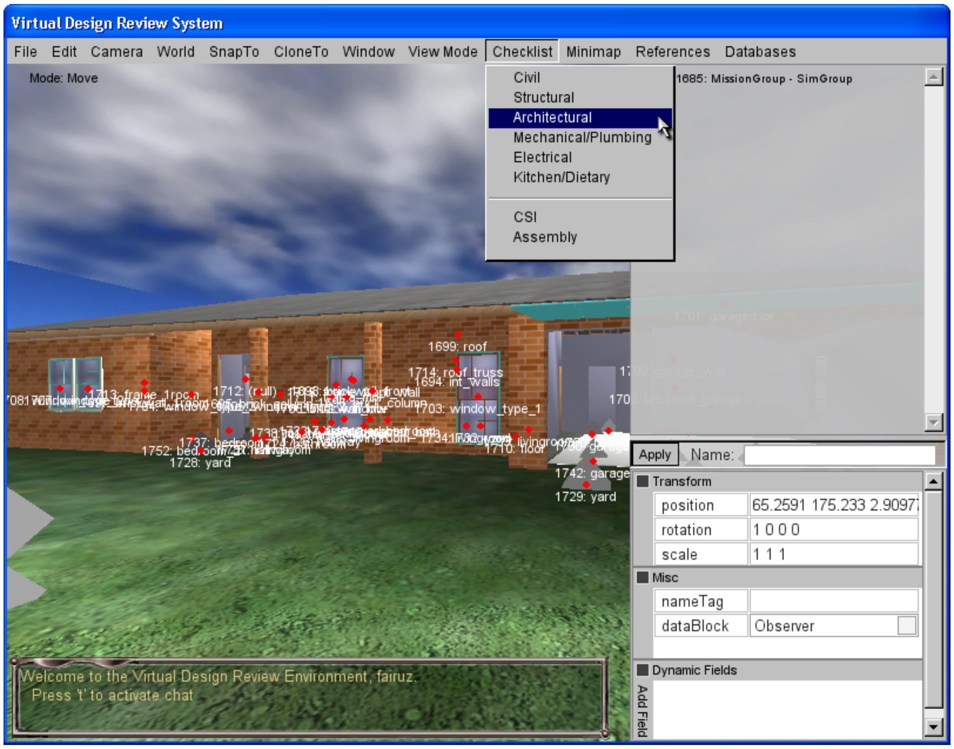


Figure 4-33 - The load checklist menu

8) The Architectural 'Task Checklist' window appears showing the first task in the checklist (Figure 4-34). The reviewer has the options to agree, disagree or place additional comments on the given task.

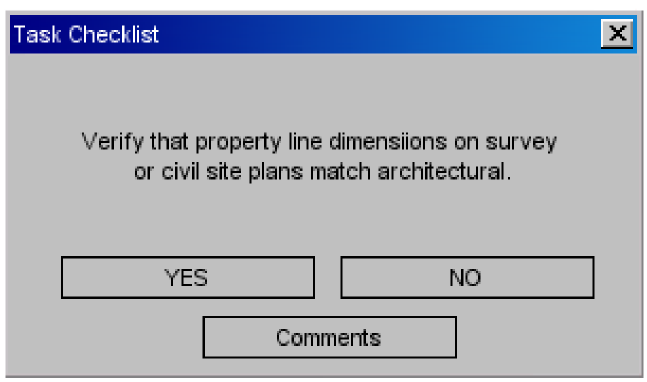


Figure 4-34 - One of the line item in the checklist

If the reviewer chooses to put in additional comment, he clicks on the 'Comments' button and the 'Review Comments' window pops-up (Figure 4-35). In the 'Review Comments' window, the reviewer types in the comments and saved it. The application stores the review comments typed in by the reviewer in the Comments database.

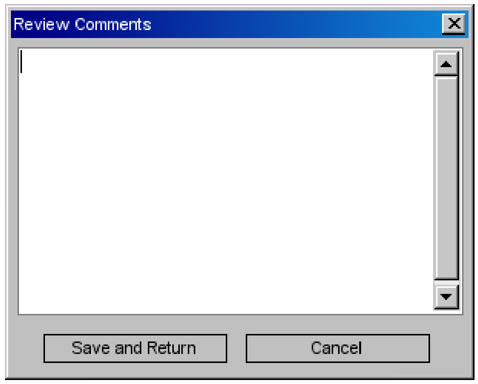


Figure 4-35 - The Review Comments window

Once a task is completed by the reviewer, the next task in line in the checklist is displayed to the reviewer in a similar fashion. The creation of the checklist allows for flexibility since the review tasks in each project differs from one to the other. Tasks can be added or removed by editing the TaskList 'Comma-Separated-Value' (csv) file. This type of file can easily be edited using any text editor or using any spreadsheet software such as Microsoft Excel. This file also dictates the rule as to how each question branches out to another question; depending on the reviewer's answers. Figure 4-36 shows an example of the TaskList.csv file opened in Microsoft Excel. The TaskList.csv file can be found under the projectOne/client/Questions/ folder.

	A	B	C
1	Thank you for using the Interdisciplinary Checklist.		
2	You have chosen to review the Civil Interdisciplinary Checklist. Continue?	Yes 9	No 1
3	You have chosen to review the Structural Interdisciplinary Checklist. Continue?	Yes 12	No 1
4	You have chosen to review the Architectural Interdisciplinary Checklist. Continue?	Yes 15	No 1
5	You have chosen to review the Mechanical/Plumbing Interdisciplinary Checklist. Continue?	Yes 18	No 1
6	You have chosen to review the Electrical Interdisciplinary Checklist. Continue?	Yes 34	No 1
7	You have chosen to review the Kitchen/Dietary Interdisciplinary Checklist. Continue?	Yes 37	No 1
8			
9	Verify that new underground utilities (power, telephone, water, sewer, gas, storm drainage, fu	Yes 10	No 10
10	Verify that existing power/telephone poles, pole guys, street signs, drainage inlets, valve boxe	Yes 1	No 1
11			
12	Verify that column grid lines on structural and architectural match. Make an overlay between tl	Yes 13	No 13
13	Verify that column locations are the same on structural and architectural.	Yes 1	No 1
14			
15	Verify that property line dimensiions on survey or civil site plans match architectural.	Yes 16	No 16
16	Verify that building is located behind set-back lines.	Yes 1	No 1
17			
18	Check type of sanitary piping under floor slabs.	Auto 19	Manual 20
19	Select the sanitary piping system under the slabs and click OK to begin.	Start 31	
20	Are the pipes made of cast iron?	Yes 22	No 21
21	Code Compliance Error: Sanitary pipes under floor slabs should be made of cast iron accordin	OK 22	
22	Are the pipes at least 2" in diameter?	Yes 24	No 23
23	Code Compliance Error: Sanitary pipes under floor slabs should be at least 2" in diameter accc	OK 24	
24	Is the pressure of the pipes at or below 100 psi?	Yes 26	No 25
25	Code Compliance Error: Pressure within sanitary pipes under floor slabs should be at or belov	OK 26	
26	Is the part number for the pipes 1234?	Yes 28	No 27
27	Code Compliance Error: The part number for Sanitary Pipes should be 1234.	OK 28	
28	Is the cost of the pipes $2.50 per foot?	Yes 30	No 29
29	Code Compliance Error: The price for sanitary pipes should be $2.50 per foot.	OK 30	
30	You have completed the Manual Compliance Checklist for the sanitary piping system under the	End 31	
31	Verify that plumbing floor plans match architectural floor plans.	Yes 32	No 32
32	Verify that new gas, water, sewer, etc. lines connect to existing or new utilites on civil drawing	Yes 1	No 1
33			
34	Verify that electrical floor plans match architectural and mechanical.	Yes 35	No 35
35	Verify that the location of light fixtures matches architectural reflected ceiling plan and that lig	Yes 1	No 1
36			
37	Verify that the equipment layout matches other discipline floor plans and that there are no coı	Yes 38	No 38
38	Verify that equipment is connected to utility systems.	Yes 1	No 1
39			
40			

TaskList

Figure 4-36 - A sample of a TaskList.csv file

9) Besides using a checklist, the reviewer also has the freedom to enquire information on any individual 3D object in the VE. E.g. if the reviewer requires more information on the brick column, the reviewer simply points to the brick column and left-click on it (Figure 4-37). The brick column will be highlighted and then a menu will pop-up with the following options to either click on 'Specifications', 'Manual Compliance' or 'Auto Compliance Check'.

Note that the 'Manual Compliance' and 'Auto Compliance Check' options are not implemented in the prototype design review application. This is one of the envisioned features where a reviewer would be able to perform codes and project specifications compliance checking. An expert-system engine will have to be utilized to develop such features.

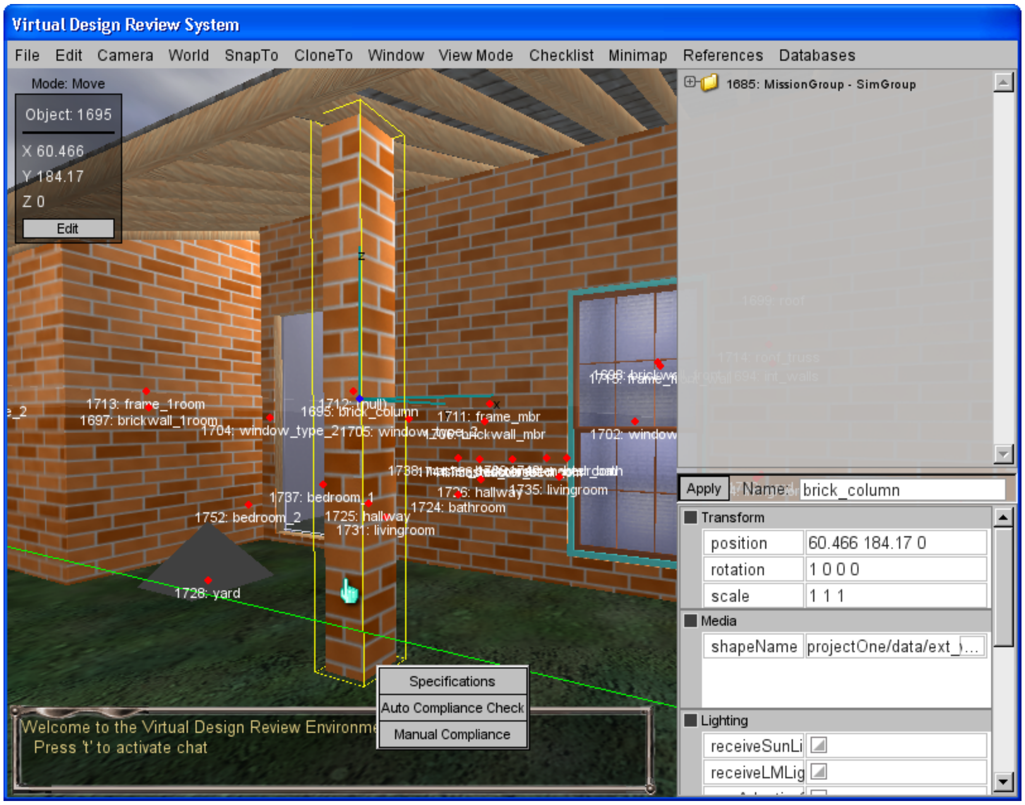


Figure 4-37 - Object selection in the VE

The reviewer then clicks on the 'Specifications' option to review the attributes of the brick column (Figure 4-38).

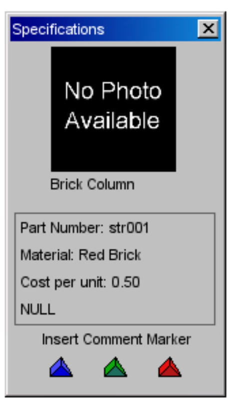


Figure 4-38 - The specifications window of the brick column

10) More information can be also accessed by the reviewer from the 'References' link on the main menu bar. The 'References' menu links information related to Codes, Manuals and Lessons Learned databases. Since the information that may be presented in these references maybe too much for the reviewers to browse through, a 'Search' feature is also available for the reviewer to search for specific keywords. Figure 4-39 shows an example of the ability to search through the 'Lessons Learned' database.

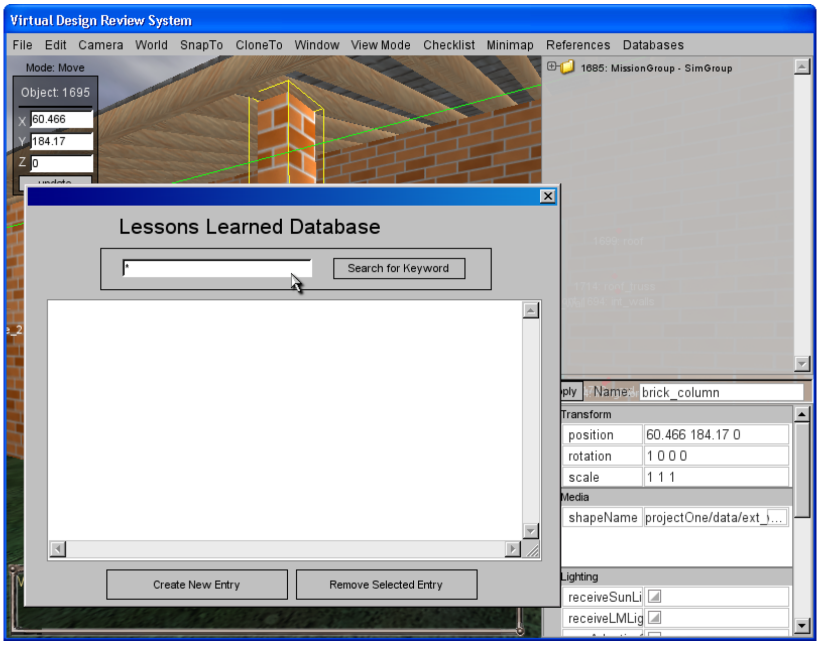


Figure 4-39 - Search window for the Lessons Learned database

11) Once the reviewer completes the review, the reviewer saves a copy of the review session for future use. The reviewer then logs out from the application.

4.3 Conclusion

The author synthesized his findings from the data gathered and analyzed in Chapter 3 to determine the required components and the related information for the design review framework. The components incorporated in the design review framework are the context-aware information processing of design review information, and a VE. A WIP prototype design review in a VE is developed to demonstrate some of the features of the framework. The WIP prototype design review application is developed using the Torque 3D Game Engine (TGE) as the enabling VE development tool.

The context-aware information processing component is responsible for filtering, retrieving, and storing design review information during a design review session. Information is stored in a collection of centralized databases (such as project specifications, building codes, standards, manuals, and etc.) therefore makes it easier for information access for design reviewers in the VE.

The context-aware rules allows for information processing of the stored design review information. The rules overcome the problem of manually processing the multitude of design review information, and facilitate data and information filter, search and retrieval. Four contexts are being used for the context-aware information filtering purposes. These contexts are: task-centric, location-centric, object-centric, and discipline centric. The rules are executed that logically link design reviewers with their discipline, location in the VE, tasks performed, and the 3D objects. This led to only relevant and needed information is presented to design reviewer in the VE.

Utilizing the context-aware rules ensures the GUI design provides design reviewer with the tools and applications to perform a task efficiently in the VE. Information is displayed in the VE in an uncrowded manner, provides clarity and readability, and avoids the unnecessary details. The GUI uses small icons or windows that do not consume much space so relationships of information can be easily recognized. The GUI displays information in logical sequencing so it eases design reviewers navigation, and information finding and access. The author also addressed the GUI principles which concern with making design reviewer to be able to feel connected/related to the visible objects displayed on the screen. The GUI design relates design reviewers to the user interface in the VE; assist on repetitive defaults (such as undo, redo, copy, and the use of visible and meaningful icons); and provides a consistent layout.

The author illustrates the GUI of the WIP prototype design review application, the information presented on the GUI in the VE, and how design reviewer can interact with the GUI. The GUI presents the design reviewer with 3D objects with embedded abstract information. Using VE technology, the VE is able to represent the owner’s intended facility in its entirety. Another advantage of using the VE technology is it allows for real-time collaboration, communication and information sharing within the same virtual 3D space. As such, in this medium, misinterpretations of design, and errors and inconsistencies in design are thus avoidable.

CHAPTER 5

5 CONCLUSION TO THE RESEARCH

5.1 Overview

The contribution of this dissertation is to provide a new framework for design review. The framework incorporates context-aware information processing of design review information in a VE. Through a work-in-progress (WIP) design review prototype, the author has demonstrated the framework has potential to improve the current drawbacks of design review. The framework promotes information centralization of the otherwise “scattered” design review information. This solves the prevailing problem of design review information being dispersed and become the hindrance for design review to be carried out more thoroughly and timely. Design review process is usually iterative because of redesigns and “hick-ups”, thus, time-consuming and resource-intensive. The author applies context aware filtering concept for the framework where information from the database is filtered, retrieved and presented visually in the VE to the design reviewer. Specifically, the framework uses Context-Triggered Actions category of context-aware. Four contexts: discipline-centric, task-centric, object-centric, and location-centric were defined and described. Under this category, IF-THEN rules are used to trigger the information processing of the required design review information and then present the information to a design reviewer in the VE.

Before embarking in this research, the author has experience in using 3D Game Engine for developing real-world application. The motivation for the author to use the Torque 3D Game Engine (TGE) from GarageGames was the clear and proven benefits of 3D Game Engine. Some of the benefits are better performance in frame-rates, the visual presentation and better integration with other technology such as database support and 3D model created using industry standard CAD software. In a 3D VE, design reviewers can visualize and interact with the 3D virtual objects, and performed unlimited virtual reviews and walkthroughs of a design. Using VE technology further improves the 3D representation of the design that can assist and provide a common language for designers and design reviewers (and other project stakeholders).

In this chapter, the author summarizes the approaches undertaken in achieving the objectives of the research. In the first section, the author sums up his motivation of doing the research that is solving the prevailing problems in the current undertaking of design review by providing a new approach of doing it. This is followed by the description of the establishment of the components for the framework. In establishing the research, the author has disseminated numerous publications in various conference proceedings and journals. The dissemination of his work has gathered valuable peer feedbacks. The author then concluded the chapter with discussion on the limitations of the research, followed by the future directions that can be explored in this particular research area.

5.2 Motivations for Research

The motivation for the author in producing this dissertation can be traced back to the underlying problems in the current undertakings of design review in the construction industry. The first problem is regarding design review information being "scattered" thus creating problems of accessing particular information at the right time, at the right place, and to and from the right person or the source of information.

From literature review on design review, the author identified various issues on design review that can be improved (Pietroforte, 1997; East, 1998, Veshosky, 1998; Baldwin *et al.*, 1999; Fu and East, 1999; Haksever, 2000; Staub-French *et al.*, 2001; Shiratuddin and Thabet, 2003a; Kubicki *et al.*, 2006). The construction industry relies heavily on manual input and output of information during design and design review. The manual inputs are prone to cause errors and inconsistencies in designs, and, and even with design review put in place, errors are still left undetected. The errors and inconsistencies in design eventually become costly to the overall budget of the project (Staub-French *et al.*, 2001). Due to this fact, there is thus a need for design review related information to be centralized, easily accessible and processed to assist reviewer during design review. The use of context-aware concept can assist the effort of providing a more centralized design review information processing as it presents only related and specific information for the respective design reviewer.

It is acknowledged that 2D design representations has been the primary conventional medium in communicating designs among project teams and conveying owner's design intentions, and it has its drawbacks (Dunston *et al.*, 2003; de la Garza and Oralkan, 1995). This problem is well-known but little or slow-effort is made to improve it such as transitioning to use more 3D design representations instead. This is largely due to the nature of the construction industry which operates with very little efficiency due to its fragmented nature; which does not allow for improved collaboration and communication in its operations (Emmitt & Gorce, 2003). Thus, there is a need to represent designs not just in 2D but 3D and in a real-time VE, where designers, reviewers, and other project team members can view and interact with the same 3D model (or 3D design).

Collaboration at the design and design review stage is valuable because collaborative work allows for the exchange of design review information. Correct decisions can be made, and errors, conflicts or other discrepancies can be detected early, thus preventing potential problems. This cooperation ensures that everyone impacted by the design has early access to design information and the ability to influence the final design early, effectively and efficiently.

Another problem is regarding the construction industry being slow to adopt new technologies in designing and representing designs, and new concepts that is intended to improve 'certain ways of doing things'. VE is one of the state-of-the-art 3D representation technologies which are underutilized by the construction industry. Other industries such as automobile and aviation have longed adopted VE technologies in their design and manufacturing processes. In a real-time 3D VE, reviewers can visualize and interact with the 3D virtual objects, and performed unlimited virtual walkthroughs of the facility. The utilization of VE technology can further enhance the 3D representation of the design that can assist and provide a common language for project stakeholders.

5.3 Establishment of the Research

The author first determined the needed information for the design review framework. To achieve this, the author performed a Content Analysis on design review documents on three of Virginia Tech's past projects: 1) the Career Services Facility building, 2) the Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech Complex, and 3) the Bio-Informatics Phase 1 building. The result of the analysis reveals there is no consistent pattern or trend to which type of error that usually occurs in design. Results also reveal that occurrence of errors/inconsistencies correlates with various uncontrollable variables such as "different designers for each project, "who designed it", "the experience of the designer", "accuracy of interpretation of the owner's design requirements" etc. Alongside the analysis, the author also gathered other form of design review related information from review of literature, interviews conducted with construction companies (this is done via the Action Research approach), publishing and presenting his work in progress in conferences.

The author synthesized all the information collected above and generated the information and its attributes to be populated in the new framework. This will serve the design review needs in a VE, the information and its attributes for 3D objects/model, and the requirements for GUI designs for the VE. Through the synthesis, repetitive information was discarded because of redundancy and irrelevance to the new design review approach. Context-Triggered Actions category of context-aware concept was used for the filtering of design review information. Four contexts: discipline centric, task centric, object centric, and location centric. Under this category, IF-THEN rules are used to trigger how the WIP prototype design review application should adapt and visually presents specific information to a design reviewer.

Figure 5-1 below shows the developed framework for context-aware information processing for design review in a VE. The framework consists of the context-aware Information processing component and the VE component.

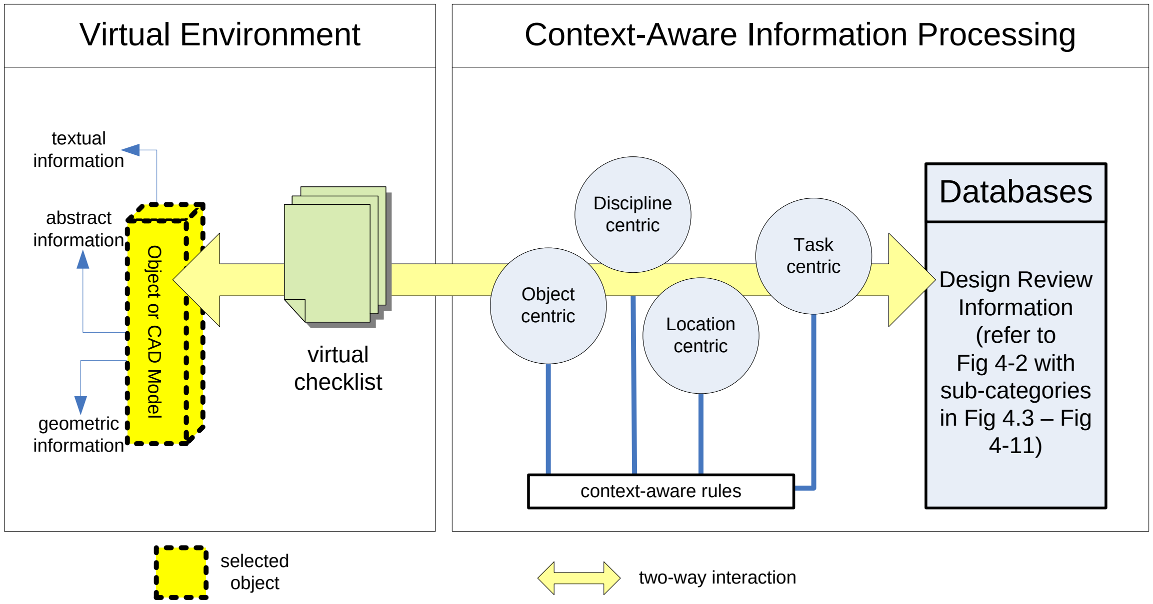


Figure 5-1 - A diagram showing the framework for context-aware information processing for design review in a VE

The context-aware information processing component is responsible for filtering, retrieving, and storing the design review information during design review. This component consists of a collection of databases where design review information is stored, and the context-aware rules that accommodate the information processing of the design review stored in the databases. The context-aware approach supports four methods of filtering information in providing and displaying the relevant information to reviewers in the VE. These contexts are:

1) discipline centric – based on the type/role of reviewer in the VE
2) task centric – based on the task performed by the reviewer in the VE
3) location centric – based on the location of the reviewer in the VE
4) object centric – based on the object selected by the reviewer in the VE

The VE component consists of 3D Objects or CAD model, virtual checklists, and information in form of textual, abstract, and geometric. The author implemented suitable information display concepts and principles of GUI design. The author refers to the works by Helander (1988), Benbasat and Todd (1993), Sears (1993), Galitz (1997), Mandel (1997), and East (1998). The WIP prototype design review application was developed using the Torque 3D Game Engine (TGE) due to the author's related work in using 3D Game Engine for real-world application. The prototype design review application was not tested for its usability, as it was not the intention of the research. However, it was developed to support the author's vision of the great potential of implementation of design review in a VE. In its current state, the WIP prototype design review application is not yet ready for real-world applications.

East *et al.* (2004), Ichida (1996), Soibelman (2003), and (Spillinger, 2000) emphasize that effective design review process produces immediate benefits. The author is confident this research has the capacity to offer such effectiveness. Properly executed design review can avoid future contingencies because design errors, omissions, and inconsistencies are detected early.

5.4 Limitations of Research

This research has the following assumptions: design review is performed using 3D models prepared by designers, and the design review application runs in a VE. Work is still in progress by design scholars and practitioners on issues of transitioning from 2D to 3D and to VE, this is a logical progression of design transitions. Designers are used to using 2D designs in their daily dealings, thus to use complex 3D tools or to use VE, requires time for training and adaptation. These kinds of changes may also involve the change in work culture and management.

This research neither directly addresses nor implemented procedures and schemas proposed by BIM and IFC. However the design review system has support for BIM's and IFC's concept of data interoperability. Future research may consider developing data structure (3D CAD model and information) that is in compliance with BIM and IFC.

5.5 Future Research Direction

This research provides the stepping-stone for the materialization of using VE for design review purposes. This framework can help defined future development requirements that can be used to guide future researchers to improve or introduce new components that may be deemed necessary in improving the design review process as technology and time progresses. Application developers can also use them as the first step to understanding the features and components required for design review to be performed in a VE.

The WIP prototype design review application developed shows a great potential in further enhancing real-time collaboration in a VE. Some of the features that may be developed are build-in support for voice-over-IP (VOIP) and video conferencing. This allows reviewers to communicate more naturally than just texting. Although the WIP prototype design review application theoretically supports up to 64 reviewers to exist in the VE at any one time, should there be a need to increase the number of reviewers, the networking codes have to be overhauled and revised.

Figure 5-2 shows a proposed future rendition of a design review system architecture in a VE. It shows the interactions between three main components, the interactions between design reviewers (and other project participants), the input, process and output of the system, and the system’s networking architecture which is based on client-server.

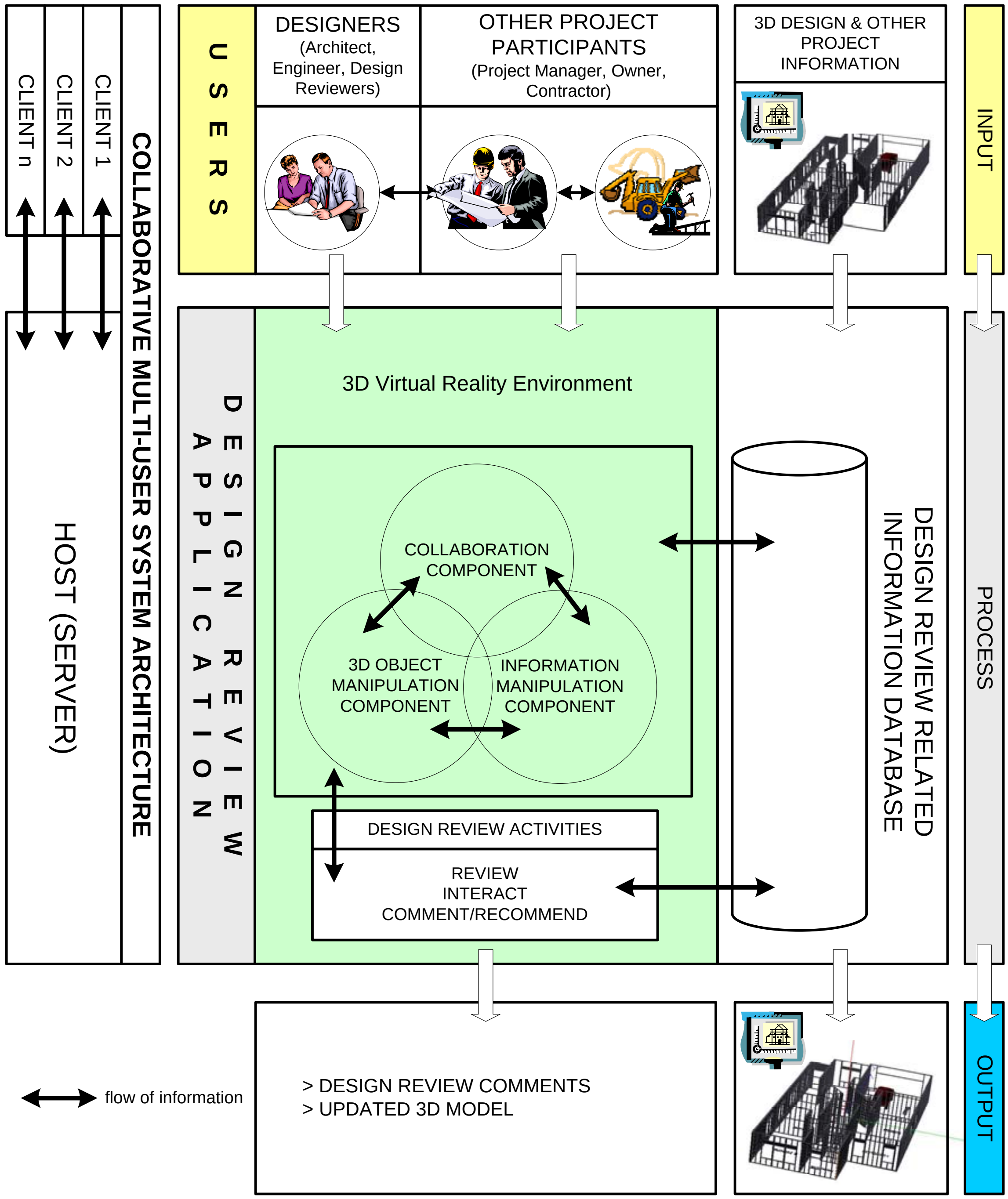


Figure 5-2 – Proposed design review system architecture in a VE for future work

With some of the components of the proposed design review system already put in place and implemented, it is hoped that future researchers will use this work as a base for further improvement and addition of new components and subcomponents. Design review is a complex subject with multidimensional variables and multitude of AEC personnel involved. Every single project is always unique in its own way and poses different challenges to project's participants. However, with a more centralized approach and processing of the required design review information, designs produced in 3D and collaboration among project's participant can be made easier. The design and design review process can ensure error-free designs, or at the very least minimizes the number of design errors and inconsistencies.

REFERENCES

Akinsola A., Dawood N., & Hobbs, B. (2000). Construction planning process improvement using information technology tools. *Proceedings of The CIB W78 2000 Construction Information Technology - Taking the construction industry into the 21st century*. Retrieved March 21, 2008 from the ITC Digital Library Website: http://itc.scix.net/cgi-bin/works/Show?w78-2000-40

Andi, and Minato, T. (2003). Representing causal mechanism of defective designs: a system approach considering human errors. *Construction Management and Economics, 21(3),* 297-305.

Aukstakalnis, S., and Blatner, D. (1992). *Silicon mirage: the art and science of virtual reality*. Berkeley, CA:Peach Pit Press.

Badler, N., Erignac, C., & Liu, Y. (2002). Virtual humans for validating maintenance procedures. *Communications of the ACM*, 45(7), 56–63.

Baldwin, A. N., Austin S.A., Hassan, T.M., & Thorpe, A. (1999, March). Modeling information flow during the conceptual and schematic stages of building design. *Construction Management and Economics, 17(2),* 155-167.

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

Barrie, D., and Paulson, B. (1996). *Professional Construction Management* (3rd ed). New York: McGraw–Hill.

Barsoum, A., Hadipriono, S.F.C., & Larew, R.E. (1996). Avoiding falls from scaffolding in virtual world. *Proceedings of The3rd Congress in Computing in Civil Engineering,* 906-912.

Behr, M, E. (2002, February 11). [Interview with Jaron Lanier, inventor of "virtual reality" term. Extreme Tech. Retrieved March 22, 2008 from http://www.extremetech.com/article2/0%2C1697%2C100970%2C00.asp

Benbasat, I., and Todd, P. (1993). An experimental investigation of interface design alternatives: icon vs. text and direct manipulation vs. menus. *International Journal of Man-Machine Studies, 38,* 369-402.

Berkoe, J. (2005). *The role of modeling and simulation in extreme engineering projects.* In National Academy of Engineering, Frontiers of Engineering: Reports on Leading-Edge Engineering, 17-22. Washington DC: National Academies Press.

Blanchard, S. B., Verma, D., & Peterson, E. L. (1995). *Maintainability: A Key to Effective Serviceability and Maintenance Management*. New York: Wiley.

Blanchard, S. B., Verma, D., & Peterson, E. L. (1995*). Maintainability: a key to effective serviceability and maintenance management.* New York: Wiley.

Bouchlaghem D., Kimmance A.G., & Anumba C.J. (2004). Integrating product and process information in the construction sector. *Industrial Management & Data Systems, 104 (3),* 218 – 233.

Bowman, D., & Hodges, L. (1997). An evaluation of techniques for grabbing and manipulating remote objects in immersive VEs. *Proceedings of The ACM Symposium on Interactive 3D graphics, 35-38.* Providence, Rhode Island: ACM Press.

Bowman, D., Kruijff, E., Laviola, J. J., Poupyrev, I. (2001, February). An introduction to 3-D user interface design. *Presence: Teleoperators & Virtual Environments, 10 (1),* 96-108.

Brooks, F. (1999). What's Real About Virtual Reality? *Proceedings of the IEEE Computer Graphics & Applications*, *19(6),* 16-27.

Bryson, S., (1996) Virtual Reality in Scientific Visualization. *Communications of the ACM, 39(5)*, 62-71.

Burdea, G., & Coiffet, P. (2003). *Virtual Reality Technology*. Hoboken, New Jersey: John Wiley and Sons.

Campbell, D.A., & Wells, M. (1994*). A critique of virtual reality in the architectural design process*. From University of Washington, Human Interface Technology Laboratory Website: http://www.hitl.washington.edu/publications/r-94-3/

Christiansson P., Da Dalto, L., Skjaerbaek J.O., Soubra S. & Marache, M. (2002). Virtual environments for the AEC sector - The Divercity experience. Turk, Z., & Scherer (eds*). Proceedings of The ECPPM 2002 European Conference of Product and Process Modelling, eWork and eBusiness in AEC*. 49-55.

Co, P. (2006). *Level design for games: creating compelling game experiences.* Berkeley, CA: New Riders games.

Cowden, J., Bowman, D., & Thabet, W. (2003). Home design in VEs. *Proceedings of The Conference on Construction Applications of Virtual Reality (CONVE).* Retrieved March 21, 2008 from the Penn State University CONVE’s website at http://www.engr.psu.edu/convr/proceedings/papers/02_Kitchens_submission_20.pdf

Dawood, N., Scott, D., Sriprasert, E., & Mallasi, Z. (2005). The virtual construction site (VIRCON) tools: An industrial evaluation. *ITcon - Special Issue From 3D to nD modelling, 10,* 43-54. Retrieved March 22, 2008 from the ITcon Website:

http://www.itcon.org/2005/5

de la Garza, J.M. & Oralkan, G.A. (1995). Using design intent for interpreting brand-name-or-equal specifications. *Journal of Computing in Civil Engineering, 9(1),* 43-56.

Dey, A.K. (2001). Understanding and using context. *Personal and Ubiquitous Computing*, 5(1), 4–7.

Dunston, P.S., Wang, X, & Lee, T. (2003). Benefit/cost analysis of desktop virtual reality application in piping design. *Proceedings of the Conference on Construction Applications of Virtual Reality (CONVE),* 22-31.

East, W. (1998). Web-Enabled Design Review and Lessons Learned. Champaign, IL: U.S. Army Corps of Engineers Construction Engineering Research Laboratories. (CERL Report No. A 396443).

East, W., Kirby, J.G., & Perez, G. (2004, April). Improved design review through web collaboration. *Journal of Management in Engineering*, 51-53.

East, W., Roessler, T.L., & Lustig, M. (1995a). Improving the design-review process: the Reviewer's Assistant. *Journal of Computing in Civil Engineering, Oct 1995,* 229-235.

East, W.E., Roessler, T.L., Lustig, M.D., & Fu, M. (1995). The Reviewer's Assistant System: System Design Analysis and Description. Champaign, IL: U.S. Army Corp of Engineers Construction Engineering Research Laboratories. (CERL Report No. FF-95/09/ADA294605).

Einstein, D. (2007, January). *A new reality: how Bechtel is using computerized virtual reality walkthroughs to help designers, customers, and the public*. Retrieved March 22, 2008 from the Bechtel, Inc. Website: http://www.bechtel.com/Briefs/0107/Virtual.htm

Emmitt, S. & Gorce, C. (2003). *Construction Communication*. Oxford: Blackwell Publishing.

Emmitt, S. (2002). *Architectural Technology*. Oxford:Blackwell Science.

Fellows, R., & Liu, Anita. (2003). *Research methods for construction (2nd ed).* London: Blackwell Science Ltd.

Finney, K.C. (2007). *3D game programming all in one (2nd ed.).* Boston, MA: Thomson Course Technology.

Foley, J. (1987, October). *Interfaces for advanced computing*. Scientific American.

Fu, Michael C., & East, E. W. (1999). The Virtual Design Review. *Computer-Aided Civil and Infrastructure Engineering, 14*, 25-35.

Galitz, W.O. (1997). *The essential guide to user interface design: an introduction to GUI design principles and techniques.* New York, NY: John Wiley & Sons.

Ganah, A., Anumba, C.J. & Bouchlaghem, N.M. (2001). Computer visualization as a communication tool in the construction industry. *Proceedings of the Fifth International Conference on Information Visualisation*. 679-683.

Haksever, A. M. (2000). A model to predict the occurrence of information overload of project managers*. Proc CIB W78 Construction Information Technology 2000. Taking the construction industry into the 21st century 2000.* Retrieved March 21, 2008 from the ITC Digital Library Website: http://itc.scix.net/data/works/att/w78-2000-415.content.pdf

Harvey, K.R. (2006). *Concrete benefits*. Retrieved March 21, 2008 from the Autodesk Inc. Website: (http://images.autodesk.com/adsk/files/Autodesk_Crom_Corp.pdf).

Heeter, C. (1992). Being there: The subjective experience of presence. *Presence: Teleoperators and Virtual Environments, 1(2),* 262-271.

Helander, M. (1988). *Handbook of Human-Computer Interaction.* New York: North-Holland Publishing.

Huang, G. Q. (2002). Web-based support for collaborative product design review. *Computers in Industry*, *48,* 71–88.

Isdale, J. (1998). *What is virtual reality*. Retrieved from September 4, 2007 from http://www.cse.dmu.ac.uk/~cph/VE/whatisvr.html

Kitchens, K., & Shiratuddin, M.F. (2007, October). Interactive home design in a Virtual Environment. *7th International Conference on Construction Applications of Virtual Reality (CONVE 2007),* 10-19. http://itc.scix.net/cgi-bin/works/Show?a893

Krippendorff, K. (1980). Content Analysis: *An Introduction to Its Methodology.* Newbury Park, CA: Sage.

Kubicki, S., Bignon, J.C., Halin, G., & Humbert, P. (2006). Assistance to building construction coordination – Towards a multi-view cooperative platform. *ITcon, vol.11.*

Kuprenas, John. (2000). *Communication growth and research needs in the future of* *construction engineering and management*. From Berkeley-Stanford CE&M Workshop: Defining a Research Agenda for AEC Process/Product Development in 2000 and Beyond. Retrieved March 21, 2008 from: http://www.ce.berkeley.edu/~tommelein/CEMworkshop/Kuprenas.pdf

Lam, H.F., & Chang, T.Y.P. (2002). Web-based information management system for construction projects. *Computer-Aided Civil and Infrastructure Engineering, 17,* 280-293.

Leedy, P.D., & Ormrod, J.E. (2005). *Practical research: planning and design* (8th ed.). Prentice Hall.

Liston, K., Fischer, M., & Kunz, J. (2000). Designing and evaluating visualization techniques for construction planning. In Renate Fruchter, Feniosky Pena-Mora, & W.M. Kim Roddis (eds). *Proceedings of Computing in Civil and Building Engineering (ICCCBE-VIII), 2,* 1293-1300.

Lok, B. (2004). Toward the merging of real and virtual spaces. *Communications of the ACM, 47(8),* 48-53.

Lombard, M., & Ditton, T. (1997). At the heart of it all: the concept of presence. *Journal of Computer Mediated Communication, 3(2).*

Lutz, James D., Hancher, D.E, & East, E.W. (1990). Framework for design-quality-review data-base system. *Journal of Management in Engineering, 6(3).*

Mandel, T (1997). *The elements of user interface design*. New York: John Wiley & Sons, Inc.

Maurina, E.F. (2006). *The game programmer's guide to Torque: under the hood of the Torque Game Engine.* Wellesley, MA: A K Peters Ltd.

Mays, P. (1998, October) Making virtual reality real. *Architecture*, 87(10), 162.

Miliano, V. (1999). *Unrealty: application of a 3D Game Engine to enhance the design, visualization and presentation of commercial real estate*. Retrieved March 20, 2008 from http://www.vsmm.org/vsmm99/

Mitchell, W. J. (1994, July). Three paradigms for Computer- Aided Design. *Automation in Construction,* 3(2-3). 239-245.

Moselhi, O. (1998). Estimating the cost of change orders. *Transactions of the American Association of Cost Engineers, 42*, EST.06.1– EST.06.5.

Mullen, M. (1998). *CGW's New 3D GameGauge. GameSpot*. Retrieved September 4, 2000 from http://www.zdnet.com/gamespot/stories/news/0,10870,2463249,00.html

Nigro, W.T., & Nigro, W.M. (1992). *Redicheck Interdisciplinary Coordination (3rd ed.).* Peachtree City, GA: The REDICHECK Firm.

Oppenheim, A.N. (1992). *Questionnaire design, interviewing and attitude measurement.* London: Pinter

Pausch, R., Proffitt, D.R., & Williams, G. (1997). Quantifying immersion in virtual reality. *Proceedings of the ACM Siggraph 97, Annual Conference Series. 13-18*. New York: ACM Press.

Pinho, M.S., Dias, L.L., Antunes Moreira, C.G., Khodjaoghlanian, E.G., Becker, G.P., & Duarte, L.M. (2002, October). A user interface model for navigation in VEs. *CyberPsychology & Behavior*, 5(5), 443-449.

Potter, N. (1989). *What is a designer? things, places, messages (3rd ed.).* London: Hyphen Press.

Poupyrev, I., Billinghurst, M., Weghorst, S., & Ichikawa, T. (1996). The Go-Go interaction technique: non-linear mapping for direct manipulation in VE. *Proceedings of the ACM Symposium on User Interface Software and Technology,* 79-80.

Pugh, S. (1990). *Total design - Integrated methods for successful product engineering*. New York: Addison Wesley Publishing.

RediCheck Associate. (2007). *Cost Savings: Study 1*. Retrieved March 21, 2008 from the REDICHECK Firm Website: http://www.redicheck-rreview.com/redichecksystem/costsavings1.html.

Redicheck. (2003). *REDICHECK Construction Document Review Service.* Retrieved March 21, 2008 from the REDICHECK Firm Website: http://www.redicheck-review.com/

Rheingold, H. R. (1991). *Virtual reality*. New York: Summit Books.

Satyanarayanan, M. (2001, August). Pervasive Computing: Vision and Challenges. *IEEE Personal Communications.*

Schilit, B., Adams, N., & Want, R. (1994, December). Context-aware computing applications. *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, 85-90.

Sears, A. (1993). Layout appropriateness: a metric for evaluating user interface widget layout. *IEEE Transactions on Software Engineering, 19(7),* 707-720.

Sheridan, T. (1992). Musing on telepresence and virtual presence. Presence: *Teleoperators and Virtual Environments, 1(1),* 120–125.

Shiratuddin, M. F., & Thabet, W. (2003b, November). Implementation issues of a design review system using VE. *Proceedings of the ASCE Construction Research Council (PhD Research Symposium).*

Shiratuddin, M. F., & Thabet, Walid. (2002, September 9). Remote collaborative virtual walkthroughs utilizing 3D Game technology. *Proceedings of the 2002 ECPPM: eWork & eBusiness in AEC.*

Shiratuddin, M.F. & Thabet, W. (2001, October). Making the transition toward an alternative VE. *Proceedings of the 2001 Conference on Applied Virtual Reality in Engineering & Construction Applications of Virtual Reality: Current Initiatives and Future Challenges (AVE II & CONVE).*

Shiratuddin, M.F. (2000a). *Utilizing 3D Game Engine to developing real-world walkthrough virtual reality application.* Master's thesis, University Utara Malaysia, Malaysia. (ISBN: T000231135).

Shiratuddin, M.F., & Thabet, W. (2002). Virtual office walkthrough using a 3D Game Engine: Special Issue on Designing Virtual Worlds. *International Journal of Design Computing (IJDC), vol. 4.*

Shiratuddin, M.F., & Thabet, W. (2003a). A Framework for a Collaborative Design Review System Utilizing the Unreal Tournament (UT) Game Development Tool. Amor R (ed). *Proceedings of the CIB W78 20th International Conference on Construction IT, Construction IT Bridging the Distance.* CIB Report 284. 310-318. Retrieved March 21, 2008 from the ITC Digital Library Website: http://itc.scix.net/cgi-bin/works/Show?w78-2003-310

Shiratuddin, M.F., & Thabet, W. (2007). Information Rich Virtual Design Review Environment. *Proceedings of the CIB W78 24th and 5th ITCEDU Workshop & 14th EG-ICE Workshop.* Retrieved March 21, 2008 from the ITC Digital Library Website: http://itc.scix.net/data/works/att/w78-2007-066-130-Shiratuddin.pdf

Shiratuddin, M.F., Yaakub, A.R. & Arif, A.S.C.M. (2000d, September). Games Engine in real world virtual reality application. *Proceedings of the 7th UK VE-SIG 2000 Conference.*

Shiratuddin, M.F., Yaakub, A.R., & Arif, A.S.C.M. (2000b, October). Utilizing 3D Game Engine to developing a real time virtual reality business application. *Proceedings of the 1st National Real-Time Technology and Applications Symposium.* (The IEEE Computer Society Malaysia Chapter, MIMOS Berhad and Universiti Putra

Malaysia, 17-18 Oct 2000).

Shiratuddin, M.F., Yaakub, A.R., & Arif, A.S.C.M. (2000c, September). Utilizing First Person Shooter 3D Game Engine in developing real world walkthrough virtual reality application: a research finding. *Proceedings of the Conference on Construction Applications of Virtual Reality: Current Initiatives and Future Challenges (CONVE 2000).*

Slater, M., Steed, A., McCarthy, J., & Maringelli, F. (1998). The influence of body movement on subjective presence in VEs. *Human Factors Journal, 40(3),* 469-477.

Smith, R.C. (2001, December). Shared Vision. *Communications of the ACM. 44(12).*

Soedarmono, D. R., F. C. Hadipriono and R. E. Larew. (1996). Using Virtual Reality to Avoid Construction Falls*.* *Computing in Civil Engineering*. 899-905.

Soibelman, L., Liu, L.Y., Kirby, J.G., East, W., Caldas, C. H., & Lin, K.Y. (2003). A design review checking system with corporate lessons learned*. Journal of Construction Engineering Management, 129(5),* 475-484.

Spillinger, Ralph S. (2000). *Adding Value to the Facility Acquisition Process: Best Practices for Reviewing Facility Designs.* Retrieved March 21, 2008 from the National Academies Press Website: http://www.nap.edu/catalog.php?record_id=9769#toc

Staub-French, S., & Fischer, M. (2001). Industrial Case Study of Electronic Design, Cost, and Schedule Integration. CIFE Technical Report #122, Stanford University. Retrieved March 21, 2008 from the Stanford University Website: http://cife.stanford.edu/online.publications/TR122.pdf

Suchman, L. (1987). *Plans and Situated Actions: The Problem of Human-Machine Communication. C*ambridge: Cambridge University Press.

Sutherland, I. E. (1965). The ultimate display. *Proceedings of the IFIPS Congress, 2,* 506-508.

Sutherland, I.E. (1963, January). Sketchpad: A man-made graphical communication system, Ph.D. Thesis. Courtesy of Computer Laboratory, University of Cambridge (January 1963 and September 2003).

Syan, C.S., & Menon, U. (1994). *Concurrent Engineering: Concepts, Implementation and Practice.* London: Chapman & Hall.

Thabet, W., Shiratuddin, M.F., & Bowman, D. (2002). Virtual reality in construction,. In Topping B.H.V., & Bittnar, Z.(eds), *Engineering Computational Technology*, 25-

52). Saxe-Coburg Publications.

The Division of Instructional Innovation and Assessment (2006). *Data Gathering: Existing Documents and Data.* Retrieved March 23, 2008 from http://www.utexas.edu/academic/diia/assessment/iar.v1/resources/quicktips/quicktip_6-50.pdf

van Dam, A., Forsberg, A.S., Laidlaw, D.H,. LaViola. Jr, J.J., & Simpson, R.M. (2000, November/December). Immersive VE for scientific visualization: a progress report. *IEEE Computer Graphics and Applications.*

Veshosky, D. (1998). Managing innovation information in engineering and construction firms. *Journal of Management in Engineering, 14 (1),* 58-66.

Vince, J. (1995). *Virtual reality systems*. Addison Wesley Longman.

virtual reality. (2007). In Encyclopædia Britannica. Retrieved June 5, 2007, from Encyclopædia Britannica Online. Website: http://www.britannica.com/eb/article-9001382

Weber, R. P. (1990). *Basic Content Analysis, (2nd. ed.)* Newbury Park, CA.

Weiser, M. (1991). *The Computer for the Twenty-First Century*, Los Alamitos, CA: Scientific American Publishers.

Whyte, J. (2000). Virtual reality applications in the house-building industry (Doctoral dissertation, Loughborough University UK, 1990)

Witmer R.G. & Singer, M. J. (2002). Measuring presence in VEs: a presence questionnaire. *Presence, 7(3),* 225-240.

Wroe, B. (1986). Contractors and computers: why systems succeed or fail. In Whyte, J. (2000). Virtual reality applications in the house-building industry (Doctoral dissertation, Loughborough University UK, 1990)

Yin, R.K. (1994). *Case study research: design and methods (2nd. ed.).* Sage: London.

APPENDICES

Appendix 1: Design Review Checklist for Construction from REDICHECK

CHAPTER 11

CONSOLIDATED REDICHECK CHECKLIST

Project Title ______________________ Project No. ______________

Reviewer's Signature ______________________ Date ______________

Note: All percentages to the right of each checklist item indicate the latest production stage at which the item should be coordinated.

1. Preliminary Review

   a. Quickly glance over all sheets, spending no more than one minute per sheet to become familiar with the project.

2. Plan Check Civil – Verify that:	Coordinated Yes	No	N/A	%
a. New underground utilities (power, telephone, water, sewer, gas, storm drainage, fuel lines, grease traps, fuel tanks) have no interferences.	[ ]	[ ]	[ ]	30
b. Existing power/telephone poles, pole guys, street signs, drainage inlets, valve boxes, manhole covers, etc., do not interfere with the new driveways, sidewalks, or other site improvements.	[ ]	[ ]	[ ]	15
c. Limits of construction, clearing, grading, sodding, grass or mulch are shown and are consistent in other disciplines.	[ ]	[ ]	[ ]	60
d. Fire hydrants and street light poles do not conflict with other above ground items.	[ ]	[ ]	[ ]	60
e. Profile sheets show other underground utilities and avoid conflicts.	[ ]	[ ]	[ ]	60
f. Horizontal distances between drainage structures and manholes match scaled dimensions and stated dimensions on both plan and profile sheets.	[ ]	[ ]	[ ]	60
g. Building footprint and finished floor elevations match other disciplines.	[ ]	[ ]	[ ]	30

Appendix 2: Examples of Lessons Learned

Lessons Learned 2

Subject: Incorrect type of toilet tissue dispensers

Project name: Career Services Building

Design Phase: Working Drawings

Discipline: Civil/Architectural/Structural & Building Code / Fire Protection
Reference: A-301

Reviewer's comment:

- "In previous drawing, 13"x14" toilet tissue dispenser were used
- According to the Uniform Federal Accessibility Standards, handicapped toilet stalls can only use the standard small roll toilet tissue dispenser (Figure 30 (d) in the Uniform Federal Accessibility Standards.)
- UFAS section 4.16.6 also requires a continuous paper flow and uses Figure 29 (b) to show the smaller toilet tissue dispenser.
- VPI & SU can only use the 13" roll as their standard in all the non-handicapped toilet stalls."

Additional Notes:
`Lessons Learned' from two previous projects were referred to:

- "On August 15, 2002, at the Substantial Completion Inspection of VPI's Lane Stadium the 13" toilet tissue dispensers had to be removed from all the handicapped toilets and the smaller toilet tissue dispensers were installed before a Certificate of Use and Occupancy was issued.
- The same problem came up on another agency's project in 1997. On November 25, 1997 BCOM received a drawing from the Department of Justice (from Ellen Harland, AIA) showing the toilet tissue dispenser could not project more than 6" into the handicapped toilet stall space."

Notes taken from UFAS (http://www.access-board.gov/ufas/ufas-html/ufas.htm)

Figure 30 (d) Grab bars complying with the length and positioning shown in Fig. 30(a), (b), (c), and (d) shall be provided. Grab bars may be mounted with any desired method as long as they have a gripping surface at the locations shown and do not obstruct the required clear floor area. Grab bars shall comply with 4.26.

Figure 29 (b) A 42 inches (1065 mm) minimum length grab bar is required to the side of the water closet spaced 12 inches (305 mm) maximum from the back wall and extending a minimum of 54 inches (1370 mm) from the back wall at a height between 33 and 36 inches (840-915 mm). The toilet paper dispenser shall be mounted at a minimum height of 19 inches (485 mm)

Lessons Learned 3
Subject: Reference to Standards
Project name: Career Services Building
Design Phase: Schematic Drawing
Discipline: Civil/Architectural/Structural & Building Code / Fire Protection
Reference: Address to Initiatives

Reviewer's comment:
"As a matter of record, it is expected the project will all inclusively address proper design/construction response to several initiatives. These include Project Program accommodation, "VT Design Guidelines and Construction Standards" (latest edition available at www.facilities.vt.edu), "Standard Specifications and Details for Site Development" (1999 ed.), "VT Cabling Standard" (October 2000 ed.)," "VT Control Strategy Standards" (May 1999 ed.), CPSM Engineering and Technical Criteria, and existing Site and Utility Constraints."

Lessons Learned 4
Subject: Reference to Standards
Project name: Bioinformatics Phase 1
Design Phase: Schematic Drawing
Discipline: Architectural/Structural
Reference: A5.1 and A5.2

Reviewer's comment:
"Consider facing generator and cooling tower concrete enclosure with Hokie Stone. This is commonly done at similar locations on campus."

Live and Snow Loadings:
"The A/E should evaluate the use of a live roof load of 20 psf and a ground snow load of 10 psf. These may not be enough, especially given the configuration of the roof and the opportunity for drifting to occur on flat sections. The university usually uses a live load consideration of 30 psf."

Basis of Design-pg. 10, Paragraph 5.a.(4).(a):
"Do not provide motion sensors in restrooms. In our experience this type of sensor does not work properly in this type of application."

Lessons Learned 5
Subject: Maintenance access for atriums
Project name: Bioinformatics Phase 1
Design Phase: Preliminary Drawing
Discipline: Architectural/Structural
Reference: Atriums

Reviewer's comment:
"As discussed in the design presentation meeting, maintenance access to high areas of the atrium needs to be carefully planned. This has been a problem area in the past."

Lessons Learned 6
Subject: Access control and security.
Project name: Holtzman Alumni Center, the Skelton Conference Center, and the Inn at Virginia Tech Complex
Design Phase: Schematic Design
Discipline: Civil Architectural, Structural

Reviewer's comment:

- **General security**

"Keying has been a chronic problem at Donaldson Brown Hotel and Conference Center for years and I know that Kelcie Edwards is justifiably very concerned with lack of key control and other issues. It is probably safe to assume that some advanced sort of access control such as card readers found in commercial lodging establishments will be used. Although this is an Auxiliary facility and as such, Physical Plant would not be directly involved in maintenance, the designer should probably involve Mr. Edwards in decisions relating to finish hardware. Physical Plant will likely be requested to provide maintenance support on a reimbursable basis, such as they are now at DBHCC."

- **Hokie Stone Wall Details**

"Wall sections relating to the installation of Hokie Stone have been an issue on the past few projects. The designer should refer to University standards, including the Harry Weese and Associates report on Hokie Stone installation guidelines developed during the last couple years in development of working drawings. Also, Mark Helms from the Virginia Tech Quarry has asked that an 8 inch space be allowed for installation of the stone veneer: this should be increased to 10 inches if the traditional 2 inch airspace between the Hokie Stone and the CMU interior wall is used. Our current thinking is that pea gravel backfill in the space is preferable to the mortar backfilling that we have been seeing on the past few projects."

- **Basis of Design Narrative Section 1.5 (Roads)**

"Agree with the designer's statement relative to the narrow, winding configuration of Duck Pond Drive. Therefore, the entrance as shown from Duck Pond to the Alumni Center may not be desirable. Additionally, suggest that designer check with VT Site/Infrastructure Department relative to the possible realignment of the intersection of Duck Pond Drive and West Campus Drive. Duck Pond Drive as it exists will NOT support heavy truck traffic from West Campus Drive to the proposed entrance off of Duck Pond Drive because of the old narrow stone bridge located near the intersection of Duck Pond and West Campus. This bridge has been hit many times in the past by tractor-trailers. Also, we agree strongly with the thoughts of Mr. Steve Mouras of VT Transportation office about the use of "B" Commuter Lot for overflow parking - this simply will not work on weekdays while Fall and Spring Semester Classes are in session. Moreover, large pedestrian movement across West Campus is probably not desirable because of the fairly heavy traffic volume present on most Fall/Spring Semester weekdays."

Appendix 3: Examples under Category 4: Sub-Categories - Errors and Recommendations

Sub-category 1: Error

Seven sub-sub-categories fall under the 'Error' sub-category. The examples of each are as follows:

Sub-sub-category 1.1: Code Compliance

Error – 1.1 Code compliance (CSF)
Design Phase: Schematics Discipline: Building Code / Fire Protection Reference: A101 Reviewer's comment: "I am concerned about the close proximity of the two exits on the south wing. Although they do meet current code, it's not by much, and (for what it's worth) it does not meet the 2000 IBC, which requires a 1/3 the diagonal separation for sprinklered buildings. Please look at reworking the stair location. If the stair is moved, it may eliminate my earlier suggestion about adding a door on the first floor southeast wall."
Design Phase: Working Drawings Discipline: Architectural Reference: Section 3.3 A Reviewer's comment: "This 13"x14" toilet tissue dispenser is not to be used in the handicapped toilet stalls. Use the standard small roll toilet tissue dispenser. See Figure 30 (d) in the Uniform Federal Accessibility Standards. UFAS section 4.16.6 requires a continuous paper flow and uses Figure 29 (b) to show the smaller toilet tissue dispenser. This same problem came up on another agency's project in 1997. On November 25, 1997 BCOM received a drawing from the Department of Justice (Ellen Harland, AIA) showing that the toilet tissue dispenser could not project more than 6" into the handicapped toilet stall space. VPI & SU can use the 13" roll as their standard in all of the non-handicapped toilet stalls only. Make correction."
Design Phase: Working Drawings Discipline: Mechanical/Electrical Reference: M101,E101 Reviewer's comment: • A/E to ensure that Mech/Elect area 159 complied with NEC 384-4(a)(1). Reference was made to the 18x62 duct shown on drawing M101.

- Reviewers could not find reference to a tear-resistance specification for roof material as required by the CPSM.

Code Reference:

- NEC 384-4(a)(1) is part of the Electrical Construction & Maintenance: Code Quandaries.
- Roofing sections against Section 707B of the CPSM.
- Provisions pertaining to roof inspector should comply with Section 707B.5
- Section 707B.6.2 specifies that the Pre-Roofing Conference should take place before materials are ordered.
- CPSM Section 707B.14.6 should be complied with regarding roof protection.

Error – 1.1 Code compliance (ACECHC)

Design Phase: Schematics
Discipline: Building Code / Fire Protection
Reference:

Reviewer's comment:

- "Current code does not allow stairwell doors to be locked from either side, and should not be relied on to provide security from floor to floor.
- Kitchen range hoods and ducts are regulated by the 1996 IMC, not by NFPA 96."

Design Phase: Preliminary
Discipline: Mechanical
Reference: Penthouse Mechanical Room

Reviewer's comment:
"The only access to all of the Penthouse Mechanical Rooms appears to be by vertical ladders through a roof hatch. This is not acceptable. Either a stairway or elevator all the way to the roof is needed in order to be able to safely and efficiently transport air filters and other materials to the equipment rooms."

Code Reference:
Design Guidelines and Construction Standards (latest revision), Division 15, General Provisions, Section 1, f.

Design Phase: Working Drawings
Discipline: Mechanical Utilities
Reference: C6.1

Reviewer's comment:
"Detail E should be revised to delete the concrete around the tee and drop outside the manhole. Backfill in this area should be stone. See the VT Design Standards for the proper detail."

Code Reference: VT Design Standards

Error – 1.1 Code compliance (BP1)
Design Phase: Schematics Discipline: Architectural Reference: A5.3: Sections 1D and 1H: Reviewer's comment: "OSHA regulations require that skylights be guarded by skylight screen or fixed railing on all exposed sides, or that the skylight be capable of withstanding a load of at least 200 pounds applied perpendicularly at any one area. Please assure that this design will meet these requirements." Code Reference: OSHA
Design Phase: Preliminary Discipline: Transportation Reference: Parking Reviewer's comment: "We will need 10 Virginia Tech standard U shaped bike racks, funded by the project. The racks should be placed on both sides of the building where cyclist will park to enter the building. These need to be shown on the drawings." Code Reference: VT Design Standards
Design Phase: Working Drawings Discipline: Mechanical Reference: S3.03 Reviewer's comment: "TYPICAL ELEVATOR PIT: The sump pit is not properly detailed. The elevator pit slab must be thickened to include concrete and reinforcing directly below the sump pit. This must be done for bidding and construction. Note: Add a removable cover to the Elevator Pit sump pit. The cover shall be fully detailed for constructability." Code Reference: Re: ASME A17.1, Rule 106.1, b, (4)

Sub-sub-category 1.2: Drawing Conflict

Error – 1.2 Drawing Conflict (CSF)
Design Phase: Schematics Discipline: Structural Reference: A202 Reviewer's comment: "(Typical Supported Floor Framing scheme, and a Typical Section showing the proposed components of the Floor System): Structural Plan is not addressed. Framed Floor System Section is addressed (See "Building Section", Sheet A202, and Narrative, "Structural" heading, item 4, "Supported Floor System")."
Design Phase: Schematics Discipline: General / Architectural Reference: Material Design Concept (page 2)- Roofing Reviewer's comment: "The sloped roof will be slate shingles with prefinished aluminum gutters and downspouts. Coordinate with Division 7 - Thermal and Moisture Protection, Roofing, Copper Gutters and Downspouts."
Design Phase: Working Drawings Discipline: Architectural Reference: A501, A504 Wall Section 2, A506 Reviewer's comment: • "Architectural precast concrete coping profile does not match the architectural precast coping shown on Scupper Detail 12 on Sheet A501 and A504. • Architectural precast concrete coping profiles do not match between wall sections and details 17 and 18 on Sheet A 506."

Error – 1.2 Drawing Conflict (ACECHC)
Design Phase: Schematics Discipline: Building Code / Fire Protection Reference: General Fire Safety Reviewer's comment: "The separation between the buildings is shown to be a fire wall in the small drawings, but not in the large set. It must be cleared that a fire wall is required to be structurally independent so that either building can collapse and the wall will be left standing."
Design Phase: Preliminary Discipline: Mechanical Reference: Basis of Design Narrative, Section 1, Pg 18

Reviewer's comment:
"Paragraph B under Central Plant Systems indicates that both chillers will be equipped with free cooling. The mechanical drawings indicate only one chiller will be so equipped. This needs to be resolved. Both chillers should have free cooling."

Design Phase: Working Drawings
Discipline: Environmental Health and Safety Services
Reference: A3.1 through A3.11

Reviewer's comment:
"Legend references smoke detectors, etc., but these are not uniformly shown all locations where indicated on electrical sheets. Please coordinate."

Error – 1.2 Drawing Conflict (BP1)

Design Phase: Schematics
Discipline: Mechanical & Architectural
Reference: M2.02 vs. A2.2

Reviewer's comment:
"The mechanical and electrical drawings follow a different numbering scheme than the rest of the drawing set (M2.02 vs. A2.2, etc.). Please coordinate."

Design Phase: Preliminary
Discipline: Electrical
Reference: C1.2

Reviewer's comment:
"The location of the switch and transformer are reversed from the positions shown on drawing E1.01. The positions on drawing E1.01 are preferable."

Design Phase: Working Drawings
Discipline: Architectural
Reference: A201B

Reviewer's comment:
"In Room 119 the size of water basin does not agree with the size of the basin shown on drawing sheets S2.01B, P2.01, P3.01, E1.01 and E2.01."

Sub-sub-category 1.3: Details

Error – 1.3 Details (CSF)
Design Phase: Schematics Discipline: Architectural Reference: Hokie Stone Veneer with Metal Stud Backup Reviewer's comment: "We wonder about the stability of the proposed structure. This stone isn't as stable as a block or brick veneer would be and the wall ties are a critical part of the assembly. Also, the A/E's detail only leaves 8 inches of clear room for the stone and air space. The stone is nominally 6" thick, but the specification for the stone allows no less than 4" and up to 8" thickness. Currently our stone is more likely to be on the 8" end of the range of allowable thicknesses, rather than the 4" thickness."
Design Phase: Working Drawings Discipline: Architectural Reference: A504, Detail 12 Reviewer's comment: "Verify that height of parapet wall matches the height of the parapet wall shown on the exterior elevations. Also, what is the material above the native stone below the scupper and behind the leader note that says "flash and seal per SMACNA recommendations"? (A/E response: Material is copper flashing over architectural precast concrete. We will delineate materials.)"
Design Phase: Working Drawings Discipline: General/Architectural Reference: #: A501, A502, A503, A505, A506 Reviewer's comment: • A/E was to comply with Division 4 – Masonry requirements, i.e. to incorporate a continuous vertical cavity with 2" rigid insulation. This was to provide a continuous "thermal break" • Reviewer found that it was critical to close the cavity at the window head and jambs for proper anchorage of the windows and to reduce the possibility of leakage. • Insulation was to be incorporated at the head, jamb, and sill conditions of Window, Louver, and Door openings (in the exterior Hokie Stone veneered wall system). • A/E was to illustrate and notate the Drainage Mat location and extent, as a separate component relating to the overall wall system at Wall Section 2/A501 (and referenced by all other Wall Sections).

Error – 1.3 Details (ACECHC)
Design Phase: Schematics Discipline: Building Code / Fire Protection Reference: Reviewer’s comment: “The Hokie stone wall design should include a drainage mat in the cavity to ensure mortar does not obstruct drainage.”
Design Phase: Preliminary Discipline: Reference: Reference “A6” series Wall Sections Reviewer’s comment: “Detail the exterior stone-veneered wall system with a continuous vertical cavity (between the rigid thermal insulation and the backside of Hokie Stone and/or Precast Concrete) from top to bottom, to ensure positive evacuation of moisture. Detail thru-wall flashing with weeps (at the head of window, louver, and door openings) below Precast Concrete elements to ensure that moisture will not stain their surfaces.” Code Reference: Division 4-Masonry, Hokie Stone, A General; Install a Drainage Mat (similar to Mortar Net, Manufactured by Mortar Net USA) continuously in the wall cavity. Exterior Sections:
Design Phase: Working Drawings Discipline: Environmental Health and Safety Services Reference: A1.3, A6.1 Reviewer’s comment: “A1.3 - Wall construction types at southwest elevator and stair are not clearly detailed. A6.1 - Section 6 shows the safety line eyebolt installation, but details here and in section 05500 are not clear: what is minimum thickness of eyebolt; what metal is allowed (stainless steel is recommended)?”

Error – 1.3 Details (BP1)
Design Phase: Schematics Discipline: Architecture Reference: A5.3 Reviewer’s comment: “Sections 1D and 1H: OSHA regulations require that skylights be guarded by skylight screen or fixed railing on all exposed sides, or that the skylight be capable of withstanding a load of at least 200 pounds applied perpendicularly at any one area. Please assure that this design will meet these requirements.”

Design Phase: Preliminary Discipline: Electrical Reference: E3.01 Reviewer’s comment: “It is noted that a ¾” conduit will be used on each end of the raceway in room 118. There are 15 data outlets shown in one of the raceways. Two ¾” conduits does not provide enough capacity to accommodate 15 cables. We suggest that the raceway be turned up vertically at each end to provide a pathway to above the ceiling. We recommend that the minimum size conduit be 1” for providing access to horizontal raceway. We also recommend that if there are more than four data outlets in a horizontal raceway, the raceway be turned up vertically as described above.”
Design Phase: Working Drawings Discipline: Architectural Reference: A2.06, Detail 5B Reviewer’s comment: “Provide approved details for the sprinklers required in the glass ceiling. Comply with NFPA 13-1994 Sections such as 4-3.1.1 and 4-4.1.5.”

Sub-sub-category 1.4: Labeling

Error – 1.4 Incorrect Labeling (CSF)
Design Phase: Schematics Discipline: Building Code / Fire Protection Reference: Reviewer’s comment: “Delete the first item under “Fire Alarm” since this is applicable to residential uses only.”
Design Phase: Preliminary Discipline: General Architectural Reference: C2, L101, P100, E401 Reviewer’s comment: • “Site Plan; Dwg #: C2; Accessible Parking Pavement marking should be noted “N.I.C” • Site Plan; Dwg #: L101; Landscaping material should be noted “N.I.C” • Plumbing; Dwg #: P100; The domestic Water Meter and Backflow Preventer should be noted “OSCI” (owner supplied, contractor installed) • Site Plan; Dwg #: E401; Site Lighting System (including fixtures, bases, conduits and conductors) should be noted “N.I.C””
Design Phase: Working Drawings Discipline: Civil/Structural

Reference: S001, S100

Reviewer's comment:
- "In the General Notes section for drawing S001, reviewer highlighted to A/E that the flat roof live load was insufficient i.e. needs to be changed from 20 psf to 30 psf.
- In the Cast-in-Place section for drawing S001 also, the reviewer highlighted to the A/E that the labels for drilled concrete piles should be changed from 4000 psi to 3000 psi. The reason for this was the former is more expensive that the latter.
- In drawing # S100, the A/E did not specifically locate a few of drilled piers by two 90 degree CL. The reviewer was concerned that the contractor might make the wrong assumption as to the location of a caisson, thus might cause potential havoc."

Code Reference:
CPSM, 701D.2. The minimum design roof live snow load for flat roofs and roofs with a slope of less than four(4) inches per foot shall be as indicated on Figure 701D-1.

Areas west of the Blue Ridge Mountains and the indicated areas of Northern Virginia shall have a minimum design roof live/snow load of 30 psf.
4000 psi concrete is more expensive because it is reinforced.

Error – 1.4 Labeling (ACECHC)

Design Phase: Schematics
Discipline: User Group Representatives
Reference:

Reviewer's comment:
- "Employee entrance, parking, restrooms (especially for kitchen employees), locker room, and break areas need to be incorporated specifically on prints.
- Meeting rooms need a numbering or naming system incorporated in the next set of prints so that they can he loaded into Delphi (our space management system) and in early 2002. our sales staff needs to begin the process of booking future group business."

Design Phase: Preliminary
Discipline: Communications
Reference:

Reviewer's comment:
"On page C1.1of the prints, the telecom duct bank is shown as two 4" conduits. This should be revised to be shown as two 4" conduits and one multi-duct consisting of two 2" and two 1.5" conduits."

Design Phase: Working Drawings
Discipline: Mechanical Utilities
Reference: Section 15685

Reviewer’s comment:

- “Paragraph 1.2: The warranty needs to be stated more clearly. The following wording is suggested:

  Special Warranty: Manufacturer’s standard form in which manufacturer agrees to repair or replace components of water chillers that fail in materials or workmanship.

  Provide equipment warranty based upon the RPM of the compressor as follows:

Compressor RPM	Warranty Term
0 – 5,000	5 years from start-up
5,000 – 10,000	5 years from start-up
10,000+	5 years from start-up plus an annual oil analysis”

Error – 1.4 Labeling (BP1)

Design Phase: Schematics
Discipline: Schematic Design Submission Narrative
Reference: Tab 3, Page 1

Reviewer’s comment:
“Jurisdiction is not Blacksburg/Montgomery County. It is in fact the Commonwealth of Virginia.”

Design Phase: Preliminary
Discipline: Mechanical
Reference: Sheet M0.01

Reviewer’s comment:
“The legend is incomplete. There are several abbreviations on the mechanical drawings that are not shown in the legend.”

Design Phase: Working Drawings
Discipline: Architectural
Reference: Reflected Ceiling Plans

Reviewer’s comment:
“The indicated bulkhead and soffit conditions shall be addressed in the sprinkler system design. NFPA 13-1994 Section 4-4.1.3 (all subsections inclusive).”

Sub-sub-category 1.5: Location

Error – 1.5 Location (CSF)
Design Phase: Schematics Discipline: Architectural Reference: Reviewer's comment: "Recommend moving the door at the file area behind the reception desk to the opposite wall so that wall space is conserved for file cabinets."
Design Phase: Preliminary Discipline: User Group Representatives Reference: A101, A102 Reviewer's comment: • "Doors in rooms 141, 142 were placed directly opposite staff lounge door and doors in rooms 143, 144 were placed directly opposite of corridor. • Doors for room 208, 209, 235 and 235 were placed directly opposite of corridor • Ethernet jack and outlet were relocated accordingly due to the repositioning of doors."
Design Phase: Working Drawings Discipline: Electrical/Fire Protection Reference: A101 Reviewer's comment: • "Elevator Machine room to be relocated to area adjacent to the elevator on first floor. Pre-action system to be moved to room 120 and floor drain to be added (see attachment; marked in yellow) • Alternatively, if elevator machine room not relocated, then pre-action system to be in a space separated from the storage room 121. • Sprinkler equipment that requires periodic maintenance/inspection is not to be located in the storage room 121." Code Reference: • Elevator code • Sprinkler code

Error – 1.5 Location (ACECHC)
Design Phase: Schematics Discipline: User Group Representatives Reference:

Reviewer’s comment:

- “There are nine handicapped rooms on the plans (we were told this was 6, what is code?) and they are not near the lobby side of the hotel. For convenience they need to be located close to the lobby and preferably more on the first floor than upper levels”.
- “Business Center on Second Level to be relocated (not convenient for guests). The business center can be relocated to the rear part of the storage area currently identified “A/V Sec. Stor. (1106 sq. ft.) facing the meeting rooms. This should have a desk front covering the opening and doors in front and rear entering into the A/V storage area.”

Design Phase: Preliminary
Discipline: Mechanical Utilities
Reference: C3.2

Reviewer’s comment:
“Why are the cooling towers located at the west end of the utility courtyard? They need to be as close as possible to the chillers. Could the tower location be switched with the outdoor electrical equipment?”

Design Phase: Working Drawings
Discipline: Environmental Health and Safety Services
Reference: A1.1

Reviewer’s comment:
“Printing room – given the large amount of combustibles, pigments/dyes, etc. utilized in this area, it would seem prudent to separate this area in a manner similar to a ‘shop’ area. Please review.”

Error – 1.5 Location (BP1)

Design Phase: Schematics
Discipline: Architectural
Reference: Sheet A2.1

Reviewer’s comment:
“As we discussed in the schematic design presentation meeting last week, relocate the entrances to the men’s and women’s toilets to the north side of the building. We recommend that these doors be in a recessed pocket which will alleviate unwanted sight lines into the restroom area.”

Design Phase: Schematics
Discipline: Architectural
Reference: A2.1

Reviewer’s comment:
“Delete extended wall which is to be the video wall and redesign to avoid dead end corridor condition in the first phase. Relocate video monitors.”

Design Phase: Working Drawings Discipline: Fire safety Reference: FP2.03 Reviewer's comment: "The portable fire extinguisher in space 334 should be relocated to space 329."

Sub-sub-category 1.6: Missing/Add/Change/Remove Item/information

Error – 1.6 Missing/Add/Change/Remove Item/information (CSF)
Design Phase: Schematics Discipline: Building Code / Fire Protection Reference: A100, A100, A101 Reviewer's comment: • "A100– I would suggest adding a door adjacent to the north stair so that egress could go directly to the exterior. • A100 – I would also suggest adding a door in lieu of the window on the southeast wall, where corridor extends between the two counselor's offices. This improves the remoteness of the available exits. • A101 – Two exits are required from the seminar room."
Design Phase: Preliminary Discipline: Building Code / Fire Protection Reference: A-504 - Detail 10/A504 Reviewer's comment: • Reviewer was concerned about the proposed wood framing for the roof. • The structure was designed in almost fully non-combustible, with the exception of wood trusses for the gable roof structure. • The use of fire retardant treated wood would incur additional cost, alternatively, use of ordinary wood would require the installation of a dry sprinkler system. • The use of non-combustible framing material, would eliminate the dry-pipe sprinkler system in the attic.
Design Phase: Preliminary Design Discipline: User Group Representatives Reference: A101 (1st floor/computer rooms & toilets) Reviewer's comment: • "The wall between the computer room and toilets to be extended to create a small alcove (bay) (see attachment: mark in yellow) • Reviewer has attached drawing/sketch showing the extension of wall to be considered by A/E.

- User wanted the wall between the computer room and toilets to be extended:
    - to improve the privacy between the toilets and the advisors' offices (no line of sight of the toilet entrance)
    - to create a small alcove which they will use in the future as a work space etc
- User preferred to equip this space with movable furniture rather than built-in shelving.
- Other requirement - A quad electrical outlet and a standard duplex voice/data jack in this space."

Error – 1.6 Missing/Add/Change/Remove Item/information (ACECHC)

Design Phase: Schematics
Discipline: Civil
Reference: Site Plan (Sheet 2 of 10)

Reviewer's comment:
"In accordance with "Accepted Cost Reductions", revise drawing to reflect the following changes:

- 3) Delete modular masonry retaining wall at the Alumni Lawn.
- 5) Delete Terrace at Restaurant (and Restaurant bump-out).
- 6), 11), 12), 13), 17) Reduce square footage/reconfigure Auditorium, Alumni Center, Hotel and Conference Center components to more closely reflect Floor Plans.
- 9) Delete the exterior stair at the Alumni Lawn.
- 10) Delete 100 parking spaces.
- 14), 16) Delete concrete stairs and sidewalls at Alumni Lawn.
- 19) Add water fountain/feature and gated entry."

Design Phase: Preliminary
Discipline: General/ Architectural
Reference: Site Plans

Reviewer's comment:
"The followings are not addressed:

- Dimensioned relationship of new work to boundaries and existing structures.
- Location of test borings
- Handicapped-accessible routes.
- Identify/show special earthwork recommended and construction considerations noted in soils report."

Code Reference:
Issues required to be addressed per CPSM, Section 807.5

Design Phase: Working Drawings
Discipline: Environmental Health and Safety Services
Reference: E2.2, A8.2

Reviewer’s comment:

- “E2.2 – If gate hold-open devices are provided in the stairs, smoke detection should also be provided in these locations.”
- “A8.2 - Details on the gates shown in stairs C2, C3, H1 and H2 need to be provided (for example, height of top rail, opening force, closing mechanism, etc. or by vendor/type with installation instructions).”

Error – 1.6 Missing/Add/Change/Remove Item/information (BP1)
Design Phase: Schematic Design Submission Narrative Discipline: Architectural Reference: Tab 3, Page 2 Reviewer’s comment: “Exterior walls envelope indicate that the building is to consist of three types: Glass curtain wall, masonry and stone veneer. Where is the glass block and what is translucent glass fiber glazing?”
Design Phase: Preliminary Discipline: Plumbing Reference: P3.01 Reviewer’s comment: “Why is there a backflow prevention device only on the water supply going to the labs? The other branch needs to have a BFD also.”
Design Phase: Working Drawings Discipline: Architectural Reference: A2.01A Reviewer’s comment: “Revise Door 110A to a double-egress door.”

Sub-sub-category 1.7: Specification

Error – 1.7 Specification (CSF)
Design Phase: Schematics Discipline: Civil/Architectural/Structural Reference: Roofing Reviewer's comment: "The preference for single ply (flat) roofs is for a 60-mil thick, fully-adhered, EPDM system. The current edition of the CPSM is the best guide for use in preparing the plans for this roof and for the slate roof. The CPSM is more current in this regard than the HECOM manual." Code Reference: CPSM
Design Phase: Schematics Discipline: Civil/Architectural/Structural Reference: Roofing Reviewer's comment: "Live and Snow Loadings: The A/E should evaluate the use of a live roof load of 20 psf and a ground snow load of 10 psf. These may not be adequate, especially given the configuration of the roof and the opportunity for drifting to occur on flat sections. The university usually uses a live load consideration of 30 psf."
Design Phase: Working Drawings Discipline: Architectural Reference: A-301 Reviewer's comment: • "13"x14" toilet tissue dispenser were used • According to the Uniform Federal Accessibility Standards, handicapped toilet stalls can only use the standard small roll toilet tissue dispenser (Figure 30 (d) in the Uniform Federal Accessibility Standards.) • UFAS section 4.16.6 also requires a continuous paper flow and uses Figure 29 (b) to show the smaller toilet tissue dispenser. • Virginia Tech can only use the 13" roll as their standard in all of the non-handicapped toilet stalls." Code Reference: Uniform Federal Accessibility Standards

Error – 1.7 Specification (ACECHC)
Design Phase: Preliminary Discipline: Environmental Health and Safety Services Reference: A4.1 Reviewer's comment: "Floor finish in the kitchen and accessory spaces is indicated to be ceramic tile (CT-5); the floor specified must provide suitable traction under wet/greasy conditions, and it is recommended that the tile be grooved or gritted. At a minimum this surface should maintain a minimum coefficient of friction of 0.5 when these surfaces are wet or greasy. Use of specialty coatings to attain this COF should be avoided unless the coating is easily maintained and has an extended use-life."
Design Phase: Preliminary Discipline: Civil/Site Reference: Specs- pg 5 Reviewer's comment: "Sewers & Sewage Disposal Systems" "D" storm sewer pipe does not have to be RCP (12" and larger). HDPE pipe is recommended."
Design Phase: Working Drawings Discipline: Environmental Health and Safety Services Reference: 09300 Reviewer's comment: "Floor tiles should be specified to provide a minimum coefficient of friction of 0.5, however, for both floor tile and terrazzo it is recommended that a minimum coefficient of friction of 0.6 be specified to comply with ADA guidelines." Code Reference: ADA guidelines

Error – 1.7 Specification (BP1)
Design Phase: Schematics Discipline: Architectural Reference: Sheet A5.1 Reviewer's comment: "Standing seam metal roof is noted to be lead coated copper."
Design Phase: Schematic Discipline: Architectural Reference: Sheet A5.3 Reviewer's comment:

“Delete Hokie stone from the interior of the building. We recommend considering other masonry materials if needed for this type of treatment.”
Design Phase: Working Drawings Discipline: Structural Reference: S7.01:Detail 9F Reviewer’s comment: “Note #3: The concrete column concrete is to be 5000 psi not 4000 psi. The note must be changed, this could be critical if the wrong strength of concrete is used.”

Sub-category 2: Recommendation

Reviewers recommended to the A/E if certain elements in the drawings should be revised and changed.

Sub-sub-category 2.1: Instruction/Advice

2.1 Instruction/Advice (CSF)
Design Phase: Schematics Discipline: User Group Representative Reference: General Reviewer’s comment: • “Reduce the size of the storage room in the staff area to about 200 square feet and add the resulting space to the workroom.” • “We suggest maximizing the glass area along the northeast side of the recruiting staff offices. If furniture placement can be accommodated, it would be very helpful to have some glass along the southeast side of the “Recruiting Staff B” office, to enable that person to help monitor what’s going on in the waiting areas and recruiting lobby area.”
Design Phase: Preliminary Discipline: General Architectural Reference: SP 100 Reviewer’s comment: “Note the entire building as VUSBC Use Group Classification B. Identify Wet Sprinkler and Pre-Action Sprinkler System protection. Reference occupancy hazard classifications and densities listed on Drawing No SP 100. Show locations of all portable fire extinguisher cabinets.”

Code Reference: VUSBC Use Group Classification B.
Design Phase: Working Drawings Discipline: User Group Representatives Reference: A-601:Finish Schedule Reviewer's comment: • "Room 115, IT Staff B Office: Flooring should be CPT rather than VCT. • Room 116, Storage: Flooring should be VCT rather than CPT. • Room 108, Director's Assistant's Office: Base should be CPT rather than RESIL. • Rooms 123, 124, 125, 126, Grad Ass't Offices and Workstations: would prefer base be CPT rather than RESIL. • Room 200, Reception, base should be CPT rather than RESIL. Will revise. • Room 107, Director's Office: What is the proposed "textured wall covering"? This is incorrect. This will be painted finish like other offices."

2.1 Instruction/Advice (ACECHC)
Design Phase: Schematics Discipline: User Group Representatives Reference: General Reviewer's comment: • "One of the communications closets on each hotel floor needs to accommodate vending (soda. ice machine that will require plumbing). The current vending space needs to be opened up to accommodate needed housekeeping storage. • There are no employee locker changing rooms or a break area. These spaces need to include bathroom facilities and lockers for 150 employees (75 in each male & female)."
Design Phase: Preliminary Discipline: Mechanical Utilities Reference: C3.2: Reviewer's comment: • "Cleanouts on the sanitary sewer lines are not acceptable. Manholes are required within 5 feet of the building exit on all laterals. • Manholes are required at all junctions. • Some of the sanitary sewer junctions are at nearly 90-degree angles. To improve flow characteristics, some of the junction manholes need to be relocated."
Design Phase: Working Drawings Discipline: Mechanical Utilities

Reference: C7.1 Reviewer's comment: "There are existing utilities (4" water line and a sanitary sewer line) that go through the Storm Water Management site. What will the cover be on the water line after finished grading? The sewer line belongs to the Blacksburg, VPI Sanitation Authority. The Authority should be contacted to make sure they are in agreement."

2.1 Instruction/Advice (BP1)
Design Phase: Schematics Discipline: Architectural Reference: A2.01A Reviewer's comment: "The 1st floor telecom room is shown. The door opens inward. The "door swing" uses almost 1/6 of the space inside the room. Please have the door swing outward. This is true for all of the CNS equipment rooms."
Design Phase: Working Drawings Discipline: Architectural Reference: A2.01A Reviewer's comment: "The discharge of the stair into the lobby is not recommended and should be revised so that the stair discharges directly to the outside."
Design Phase: Working Drawings Discipline: Fire Protection Reference: A2.06, Detail 5B Reviewer's comment: "Provide approved details for the sprinklers required in the glass ceiling. Comply with NFPA 13-1994 Sections such as 4-3.1.1 and 4-4.1.5."

Sub-sub-category 2.2: Potential Conflict

2.2 Potential Conflict (CSF)
Design Phase: Schematics Discipline: Civil/Architectural/Structural & Building Code / Fire Protection Reference: Reviewer's comment:

- Was concerned on the stability of the structure.
- Insisted that the Metal Stud Backup/GWB (gypsum wall board)/drywall backing was not as stable as block or brick veneer.
- The use of wall ties is a critical part of the assembly; Wall ties, Hokie Stone and GWB will not work well.

Reviewer (b):

- Was concerned about the similar issue as (a).
- Raised the issue of moisture migration problems in the past with Hokie stone, he was certain that such problem will not occur with CMU back-up wall.
- Agree with the use of GWB only if it was water resistant.

Design Phase: Schematic
Discipline: Civil/Architectural/Structural
Reference: Service Access

"Although we look to Steve Mouras (Transportation) for more detailed comments regarding parking and transportation issues, we have serious reservations about the adequacy of parking and access for service vehicles in the new quadrangle created by this building, the proposed Student Services Building, and the proposed New Residence Hall. The entire situation is aggravated badly by the fact that the area of West Campus Drive and Washington Street is already badly congested and that the crossing between (near) the proposed site and Litton Reeves Hall is one of the most heavily traveled pedestrian routes on campus."

Design Phase: Working Drawings
Discipline: Civil/Structural
Reference: S001, S100

Reviewer's comment:

- "In the General Notes section for drawing S001, reviewer highlighted to A/E that the flat roof live load was insufficient i.e. needs to be changed from 20 psf to 30 psf.
- In the Cast-in-Place section for drawing S001 also, the reviewer highlighted to the A/E that the labels for drilled concrete piles should be changed from 4000 psi to 3000 psi. The reason for this was the former is more expensive that the latter.
- In drawing # S100, the A/E did not specifically locate a few of drilled piers by two 90 degree CL. The reviewer was concerned that the contractor might make the wrong assumption as to the location of a caisson, thus might cause potential havoc."

A/E Response:

- A/E complied and made changes

Code Reference:

- CPSM, 701D.2. The minimum design roof live snow load for flat roofs and roofs with a slope of less than four (4) inches per foot shall be as indicated on Figure 701D-1.

- Areas west of the Blue Ridge Mountains and the indicated areas of Northern Virginia shall have a minimum design roof live/snow load of 30 psf.

2.2 Potential Conflict (ACECHC)

Design Phase: Schematics
Discipline:
Reference:

Reviewer's comment:

- "There are design and operational concerns with the 3rd floor meeting rooms regarding noise and vibrations from air handlers and cooling equipment of the floor above them. Also possible exhaust problems with kitchens below."
- "Beds are shown on the plans as flush to the wall. This presents a challenge for housekeepers as the beds cannot be efficiently made and is not consistent with a four-diamond facility."

Design Phase: Preliminary
Discipline: Mechanical Utilities
Reference: Sheet C3.1, Sheet L2.2

Reviewer's comment:

- "Sheet C3.1 - Why is the underground waterline on the north side of the facility configured with so many bends? It seems that this configuration adds many unnecessary fittings that add to the cost and potential future maintenance problems".
- "Sheet L2.2 - There appears to be trees planned in close proximity to the cooling towers. This may cause maintenance problems".

Design Phase: Working Drawings
Discipline: Fire Protection
Reference: C, L, A, & FP

Reviewer's comment:
Exit Discharge:
"The Exit Discharge, by definition, extends from the termination of the exit to a public way. Obviously, the efficient movement of the occupants from the building to a point of safety from a fire is necessary. Simply being outside of the building is not sufficient to avoid the deleterious effects of a burning building. If all occupants cannot continue their uninterrupted travel to a point of safety, the Means of Egress continuity is incomplete.

Define which Exits are accessible by all occupants. Demonstrate Compliance for these Exits with Section 1006.3 "Exit Discharge" and "Continuity of the Means of Egress" to the Public Way. Demonstrate that the surfaces encountered are accessible by all occupants. For those Exits that are not accessible by all occupants, define how the "Safe Egress" of the remaining occupants is achieved. This issue was first discussed during the

June 12, 2002 meeting at BCOM.”

2.2 Potential Conflict (BP1)
Design Phase: Schematics Discipline: Mechanical Reference: M2.03 Reviewer’s comment: • “Coordinate location of atrium smoke evacuation exhaust fans with the skylight and design.” • “A general note: Provisions for fall protection for personnel working on rooftops will become an issue. Fall protection must be incorporated into the design of the project; however, the type of fall protection needs to be solved in such a way as to minimize or eliminate visual impact.”
Design Phase: Preliminary Discipline: Mechanical Reference: M2.02 and M2.03 Reviewer’s comment: “Two 40” X 30” ducts are shown being routed through telecom rooms 233 and 321. We are concerned about the installation height of these ducts and their possible negative impact on the amount of usable space in those rooms. The cable tray typically is routed into the telecom rooms at a minimum height of 8’ and we will need some clearance above that the top of the cable tray to accommodate routing of the cables. At what height will these ducts be mounted?”
Design Phase: Working Drawings Discipline: Civil/Structural Reference: Project Manual (03310, page 5) Reviewer’s comment: “Delete the use of fly ash and add a prohibition statement against their use for exterior concrete applications. They shall not be used as a substitute for the required cement to meet the strength requirements of the mix. Fly Ash increases the cure time, reduces durability and reduces the effectiveness of the air entrainment admixtures. See ASTM C-618. Interior concrete applications - See discussion of use of fly ash in exterior concrete applications in comment above. Applicability to other concrete is problematic due to the low early strength and reduction of effectiveness of the air entrainment admixture (if used). It is recommended that it also be prohibited throughout, but if allowed, it shall not be used a substitute for any of the required cement content to yield the required mix design strength. There may be some benefit of its use as an admixture for long term

added strength and to reduce the porosity of the concrete, but care is required to adjust the mix to accommodate its addition. Please revise the specs to provide the proper instructions and admonitions, if fly ash is allowed."

Code Reference:
ASTM C-618.

Appendix 4: Categories of Design Review Information for the Design Review Framework (refer to Appendix_4.pdf file)

Appendix 5: Prototype Development

5 Development of Prototype Design Review Application in a VE Environment

5.1 Introduction

In this Appendix, the author explains the use of GarageGames' Torque Game Engine (TGE) version 1.4.2 to develop a prototype design review application for this research. Features for this prototype include real-time 3D and information visualization and real-time collaboration capability in a VE across the network. The author also describes and illustrates user interaction with the prototype design review application.

The TGE was originally developed by a computer game development company called Dynamix. Dynamix created computer games titles such as Earthsiege, Starsiege, Tribe and Tribes 2 (Maurina, 2006; Finney, 2007). The founders of Dynamix then created GarageGames and currently distribute the TGE as one of the products for independent (indie) game developers. The low-cost licensing option has been attractive enough for many indie developers to use the TGE for development purposes. At the time of research, the TGE was purchased based on an indie license for $150. GarageGames' business model is somewhat different from many other 3D game engine developers. Not only the licensing cost is affordable, the license also includes the entire TGE source code. The TGE is a commercial grade game engine which has been successful in bridging the gap among multiple industry sectors (Shiratuddin & Fletcher, 2006).

Besides the TGE, based on the author's experience with other 3D game engines such as Epic Games's Unreal engine and Valve's Half-Life engine, it is concluded that without access to the entire engine's source code, further modifications of the engine to support real-world applications such as design review are impossible (Shiratuddin & Thabet, 2003a). The author has worked using EpicGames's Unreal engine, Valve's Half-Life engine, id Software's Quake engine, and some other commercially available game engines to develop the prototype design review application. However, none of these engines came with the entire source code, thus limit the engines from being extended further to include features for a design review application.

During the investigation stage of the research and prior to the development process, the author developed Figure 1-1 which shows the components and structure of the TGE that were used for the prototype design review application. Although documentation on the TGE is plentiful and can be found on GarageGames' website, most of the documents were old and did not apply directly to the development of the prototype design review application. In the investigation, the author found that some of the documentation provided on GarageGames' website on the TGE were redundant and no longer applies to current releases of the TGE. This caused difficulty in understanding and finding the specific codes or functions. Some of the TGE codes spanned across multiple files in different folders therefore making the development process challenging. However, the TGE online community has provided assistance to the author. Finney (2007) and

Maurina (2006) provide information on the TGE. In the author's opinion, a more complete and up-to-date documentation of TGE is an area that requires attention and improvement from GarageGames and its community. Redundant information should either be updated or removed to avoid confusion.

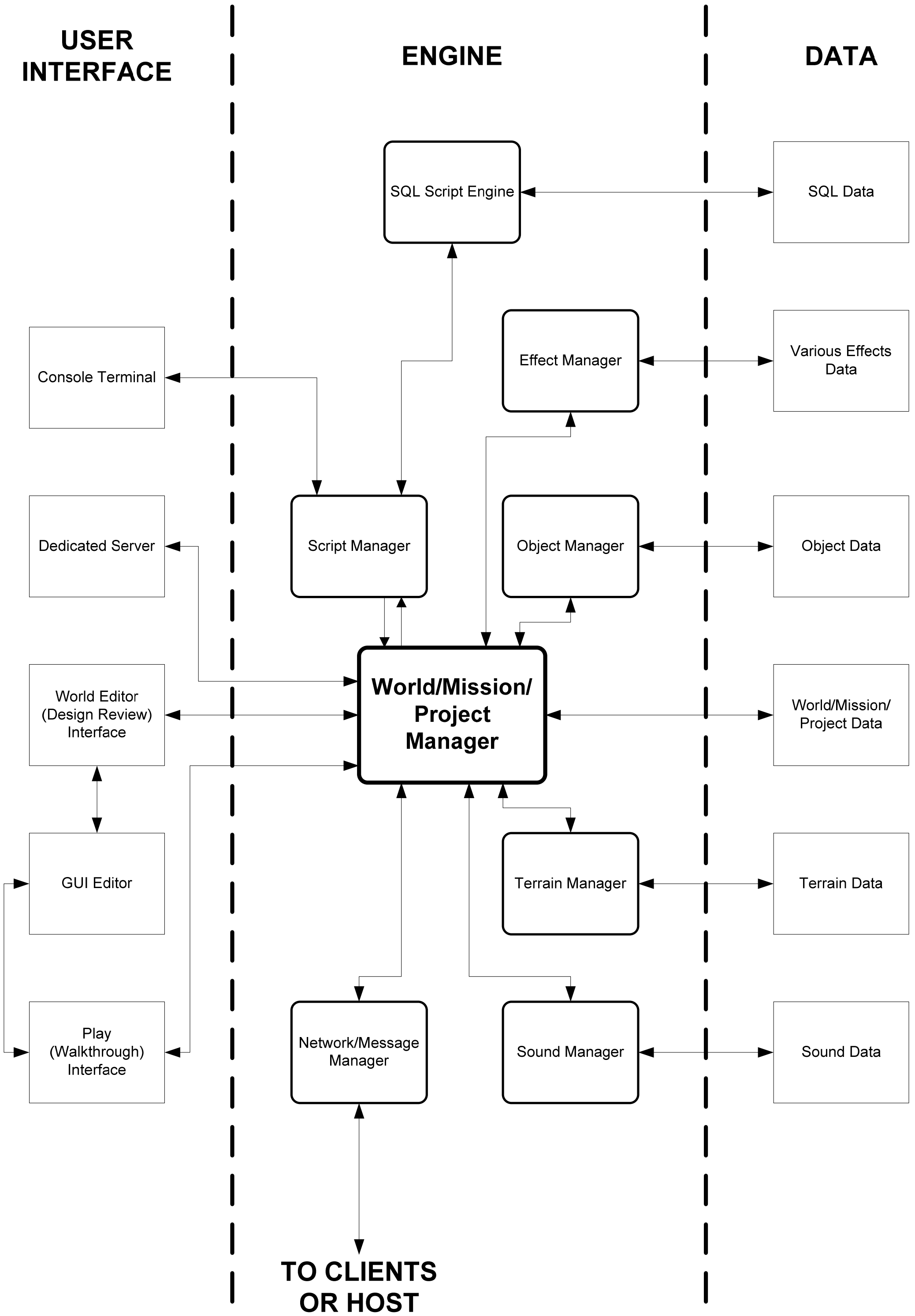


Figure 1-1 - The components and structure of the TGE applied to the prototype application.

3 main components of the TGE are:

1) The **User interface** (UI) – this component consists of several sub-components i.e. the **Console Terminal**, **Dedicated Server**, the **World Editor**, the **GUI Editor** and the **Play** mode. The UI component comprised of both text and graphical-based UIs. The UI allows for interaction between the application and the end-user.

   The **Console Terminal** sub-component uses only text-based interaction. The console terminal can be of two types: (1) a text chat window and (2) the developer's debugging terminal. While each type represents different functionality, both used text-based interaction with the end-user. The **Dedicated Server** sub-component allows for setting up and launching a dedicated standalone server using the **Console Terminal**. A dedicated server is usually setup to wait and listen, and link any incoming and outgoing connections from external clients. The server also synchronizes all the data between the clients and **World/Mission Manager**. The next UI sub-component is the **World Editor** or **Design Review** interface. The **World Editor** is where the visual scene assembling and design reviewing in the VE occurred. The **World Editor** employs a more graphical approach to many of 3D object manipulation functionalities such as moving, deleting, rotating, scaling etc. Besides 3D object manipulation, the **World Editor** is closely linked to the GUI Editor. The GUI Editor also employs a visual approach in GUI design. It allows for the creation and modification of UIs for the design review prototype application such as adding new menu items, right-click menu, the login screen etc.

   The **Play** sub-component allows for real-time viewing of the VE. In the **Play** mode, only real-time walking-through the VE is allowed. No object or information manipulation exist is this mode. This mode is suitable for users who would like to only view the 3D model with no intention of making changes or comments to the design.

2) The **Engine –** this is the main component or the heart of TGE. The **Engine** component consists of various managers. Each manager is responsible to handle, retrieve and process specific type of data. At the center of the **Engine** is the **World/Mission Manager**. The **World/Mission Manager** manages all the interaction among managers before passing the final required data to the end user.

   Table 1-1 shows descriptions of the various managers that exist in the Engine component.

Manager	**Description**
SQL Script Engine	Manages information storage and retrieval
Effect Manager	Manages visual effects such as displaying the skybox, textures, lightings, highlights around an object when selected etc.
Script Manager	Manages the execution of any scripts

	required by other managers and then passes it to the World Manager and Console Terminal
Object Manager	Manages all the objects in the VE which includes 3D objects, 2D GUI, avatars etc. Also manages properties such as texture mapping, geometry size, GUI size and color etc. Note that any items present in the VE is treated as object.
World/Mission/Project Manager	Tracks the location of each object present in the VE. All data must go through the World Manager and then redistributed accordingly to other managers or to the end-user.
Terrain Manager	Manages the rendering of terrains. The terrain engine is part of the terrain manager and it is a sub-set of the main engine.
Network/Message Manager	Manages any data that requires updating across the network. Whenever real-time collaboration occurs, any changes made by any reviewers will be relayed to the hosting server and then retransmitted to other clients in the network.
Sound Manager	Manages all sound related data such the sound effect when an avatar is walking in the VE.

Table 1-1 - Description of the engine's managers

3) **Data –** various data is stored and retrieved in different data elements. The required data for the design review prototype application are SQL data, various effects data, object data (textual and 3D object), world mission data, terrain data and sound data. E.g. Various Effects Data stores and retrieve effects such as glow, display of bounding boxes, lightings, shadows etc. The SQL data element stores and allows for retrieval of objects information and design review information. Table 1-2 describes the various types of data stored in the data component.

Data	**Description**
SQL data	Stores information in databases in the SQL format. Information stored here includes log files for each design reviewers, 3D objects information, design reviewers' comments, lessons learned etc.

Effects data	Stores visual effects related data such as the skybox, textures, lightings, highlight color around an object when selected etc.
Object data	Stores data of all the objects in the VE that include 3D objects, 2D GUI, avatars etc., and their properties such as texture mapping, geometry size, GUI size and color etc.
World/Mission/Project data	Stores the world/mission/project files so they can be retrieved later.
Terrain data	Stores terrain data such as heightfield elevation map, the terrain geometry, and terrain textures.
Sound data	Stores sound data that may include sound effects such as footsteps, weapon fire, weapon damage, background music, etc.

Table 1-2 - Description of the types of data

Another important element of the prototype design review application is scripts. The 3D engine provides the code for graphics rendering, texturing, etc, and these capabilities are tied together by scripts. Scripts are often used to bring different parts of the engine together, to provide a functional VE.

Below is an example of a script used in TGE for the prototype design review application. The excerpt is taken from the “main.cs” script file. The scripts shown below changes the main window’s title to “Virtual Design Review System” on the execution of the main executable file. This “main.cs” resides in the /projectOne/ folder.

During the development process, the author made the effort to provide a readable and easy to understand scripting notation. Some of the lines of scripts were documented with comments written at the end of each line.

```
Redacted….[GarageGames Copyrighted material]

……
……
// InitCanvas starts up the graphics system.
// The canvas needs to be constructed before the gui scripts are
// executed because many of the controls assume the canvas exists at
// load time.

initCanvas("Virtual Design Review System");    // initializing the application Window Title and named it as Virtual Design Review System. The default name was "TORQUE TUTORIAL BASE"

// Load the client-side Audio Profiles/Descriptions
exec("./client/audioProfiles.cs");
……
……

Redacted…[GarageGames Copyrighted material]
```

5.2 *Prototype Design Review Application*

The prototype design review application is a modified version of the TGE. Using C++ and C-like syntax scripting language, additional codes were written to provide the added functionalities specific for design review purposes. Microsoft Visual Studio 2005 was used as the programming IDE (integrated development environment) or compiler, installed on an IBM-PC compatible desktop computer. The prototype design review application has been tested and supports VE devices such as the head-mounted-display (HMD), tracking devices, data glove, 3D navigation input devices, game input devices and stereoscopic display (using nVidia consumer stereo display driver).

Figure 1-2 shows the building blocks of the prototype design review application and how the blocks are linked to one another. The TGE provides the real-time 3D visualization in the VE, the graphical user interface (GUI) and the capability for 3D object manipulations. Whenever a design reviewer interacts with the 3D objects and information in the VE, interactions among the three building blocks occur in the background. The interaction between the database engine and rule-based engine is almost cyclical. When the requested information has been retrieved (based on the flagged ***True*** statement provided by the rule-based engine), the information is then passed to the TGE so the information can be visually presented to the design reviewer in the VE.

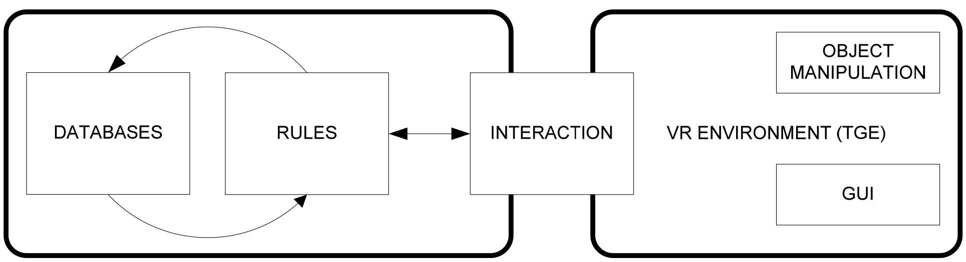


Figure 1-2 - The building blocks of the prototype design review application

In the prototype design review application, 2D drawings are replaced by 3D models in a VE. The 3D models, checklists, and sources of design review related information are centrally located and stored in databases, which can be accessed by the design reviewer at a click of a button in the VE. Access to design review related information is benefited from a defined context-aware search and retrieval processes.

The context-aware search and retrieval processes include filtering, querying and sorting the required design review information. The processes are based on four information processing contexts: D, T, O, L (see Chapter 4). Whichever mode selected by the design reviewer, and once the 3D model in the VE has been reviewed, if any errors or inconsistencies are encountered, the design reviewer can quickly access any related design review information stored in the databases for validation. After verifying the design issues, the design reviewer can then input any comments or recommendations into the design review system. The comment or recommendation is then stored in the respective database and is accessible to other design reviewers, and most importantly to the designer who is responsible to make the design revisions.

The development of the prototype design review application involves two major phases:

1) development of VE assets e.g. the 3BR House 3D Model and textures
2) modification of existing or the introduction of new functionalities through programming such as the implementation of real-time 3D object manipulation, real-time collaboration across the network, GUI, information processing and visualization

5.3 Prototype Development

This section describes the development process of the prototype design review application. The two major development phases were 1) creating the assets i.e. the 3BR house model and its matching textures, and the assembly of the VE, 2) modifying existing functions or introducing new functionality through programming to support design review in a VE.

The prototype was developed in a period of approximately 1-year. In streamlining the development process, only certain tasks were undertaken at any particular time. These tasks include the preparation of the 3D model, designing and programming the GUI, improving 3D object manipulation functionality, introducing real-time collaborative design review across the network and embedding database support for information processing.

Through a series of phone and face-to-face interviews and meetings with a local architect, inputs and feedbacks were gathered on the functionality and the look-and-feel (focusing more on the GUI design and its functionality) of the prototype design review application. Although, many of functionalities were suggested by the local architect, some were not implemented because of the complexity of the programming with TGE, or simply TGE does not support such extensions.

Figure 1-3 shows a breakdown of the development process and, the interaction between the author and the local architect.

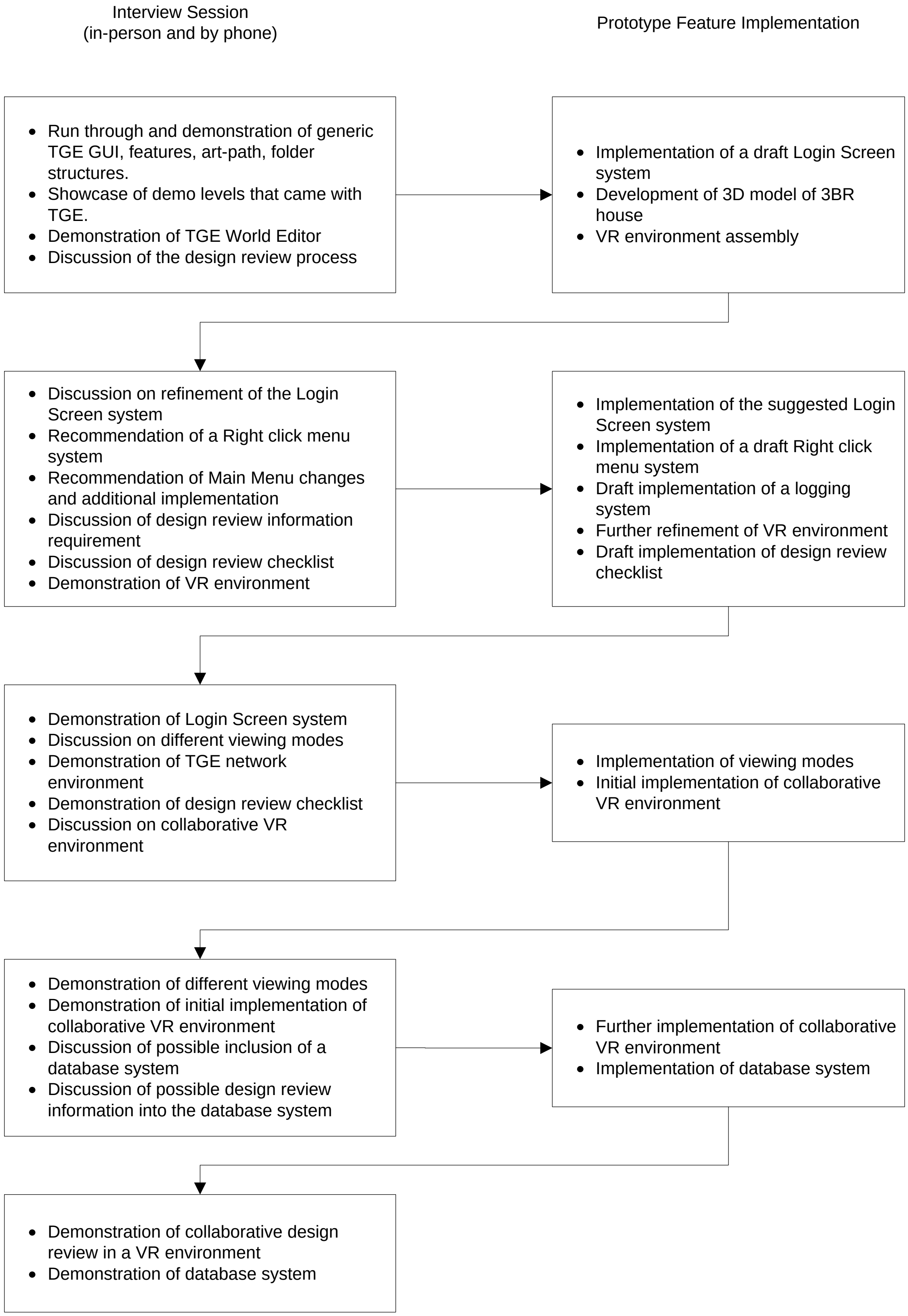


Figure 1-3 - Interviews conducted to assist development of the prototype application

5.3.1 VE Scene Assembling: Creating the 3D Model and Textures of the 3 Bedroom House

In this development phase, three main steps were involved. Step 1 was the modeling of the 3BR house and preparation of the matching textures for all the 3D objects that made up the house. Step 2 was the placement of all the 3D elements and matching textures into specific TGE data folders. Step 3 involved the assembling of the VE Environment. A summary of the steps is shown in
Figure **1-4.**

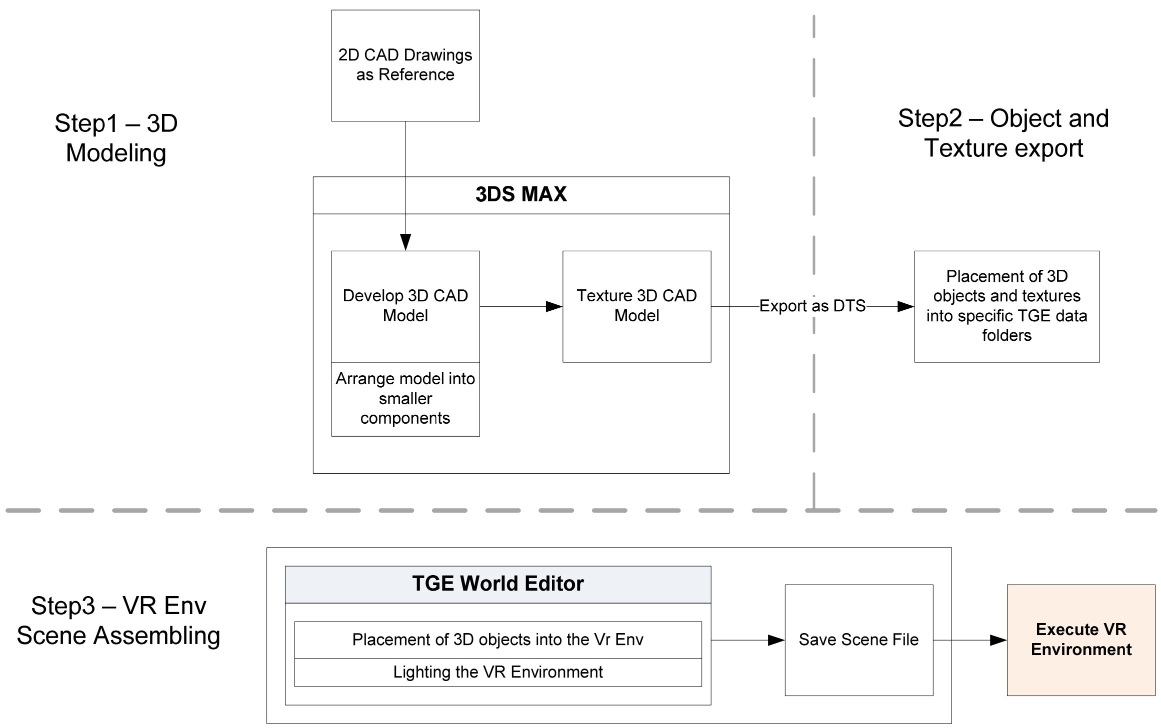


Figure 1-4 - The steps involved in transferring a 3D CAD model into the prototype design review application

Step 1: Developing the 3BR House 3D Model

Based on a 2D CAD floor plan drawing, the 3BR house 3D model was created using Autodesk 3DS Max 9.0 3D modeling software. The house had to be divided into several sections. For example, similar 3D objects are grouped into distinguishable elements e.g. exterior walls, doors, windows etc. This step is important to avoid crashing the 3DS Max's TGE DTS exporter. Dividing the 3D model into smaller sections also helped to manage 3D objects more easily. A 3D modeling layer management approach was used and this approach is similar to any 2D CAD design where different types of elements should be placed in different layers (Figure 1-5).

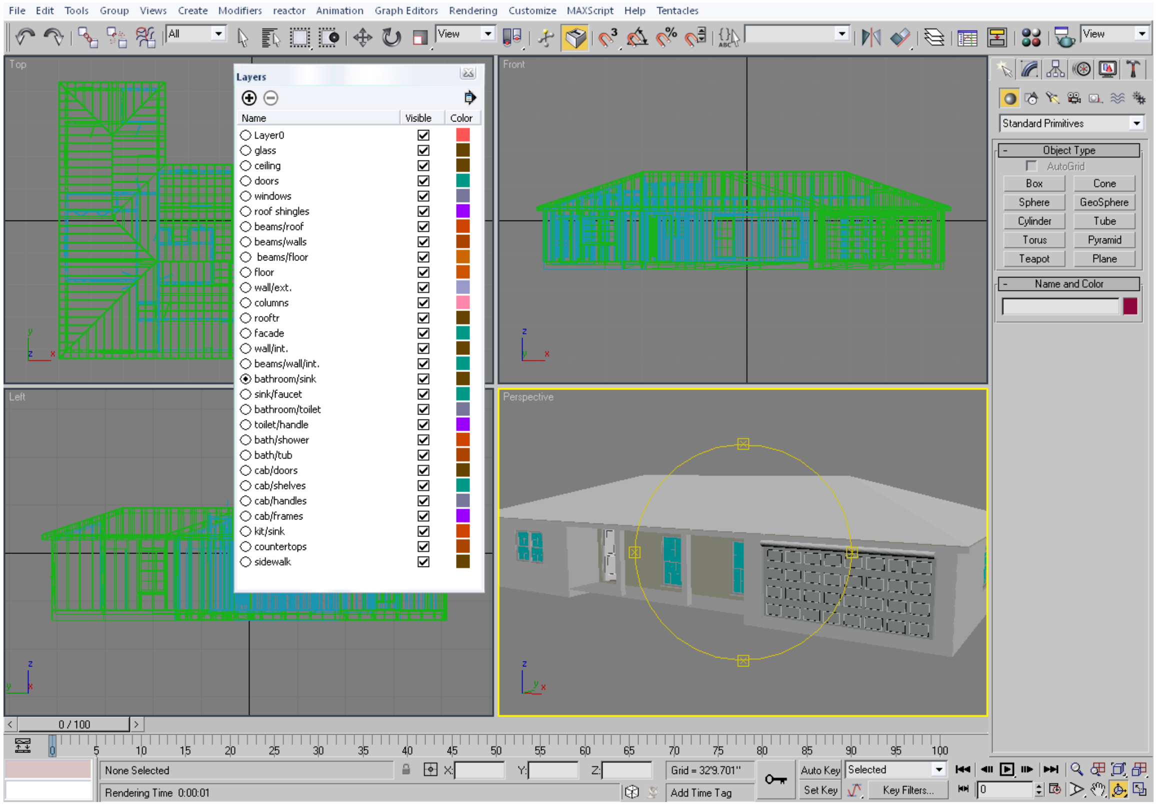


Figure 1-5 - Layer management approach for the 3D BR house model

Once the 3D modeling process was completed, various textures were applied accordingly to all 3D objects. The textures must either be in the JPG (JPEG – Joint Picture Expert Group) or PNG (Portable Network Graphics) image file format since these are the only texture file format supported by the TGE. Textures were created using a texture software creation application called MapZone developed by Allegorithmic (Figure 1-6).

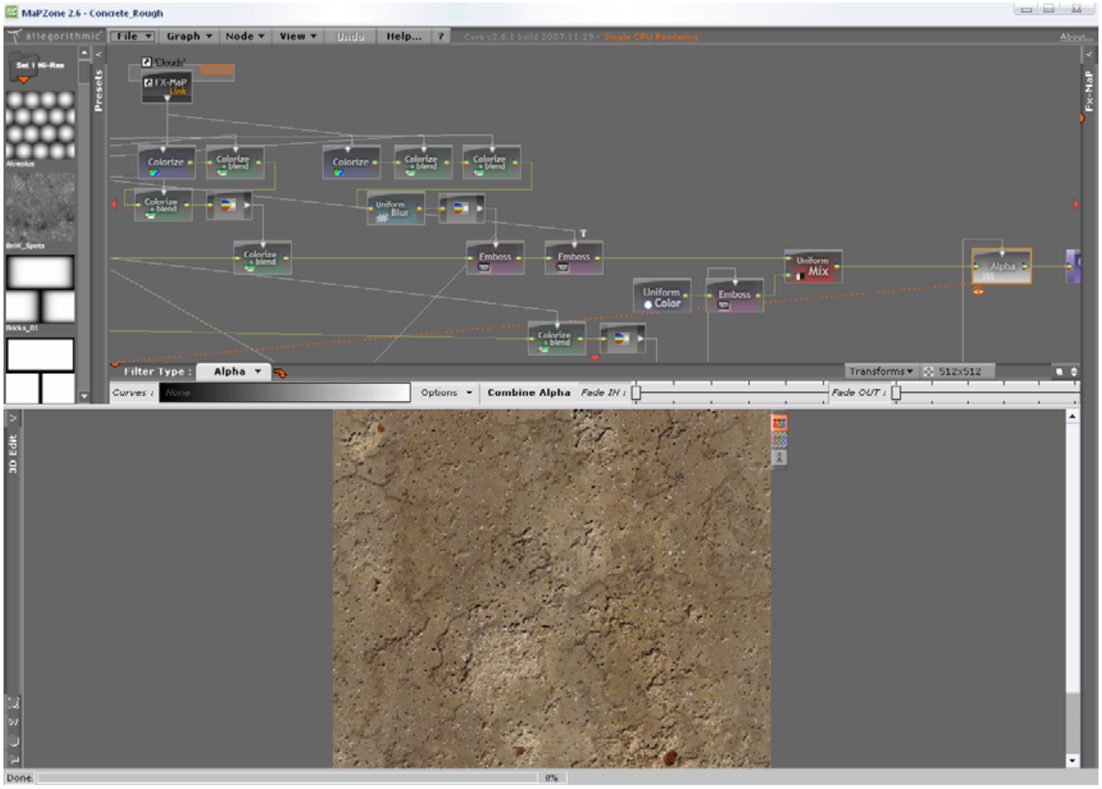


Figure 1-6 - Sample screenshot of MapZone

Before exporting the 3D objects of the house into the TGE DTS file format, the 3DS Max to DTS exporter plugin must be installed first since by default, 3DS Max does not come with the exporter. The DTS exporter was downloaded from GarageGames website. Once the exporter plugin was installed, a utility icon  was created on 3DS Max main menu bar for easier access. The DTS exporter plugin made it easier to export 3D object into the TGE DTS file format (Figure 1-7). The 3BR house's 3D objects were exported one group at a time into the DTS file format.

Figure **1-8** shows rendered images of the 3BR in 3DS Max.

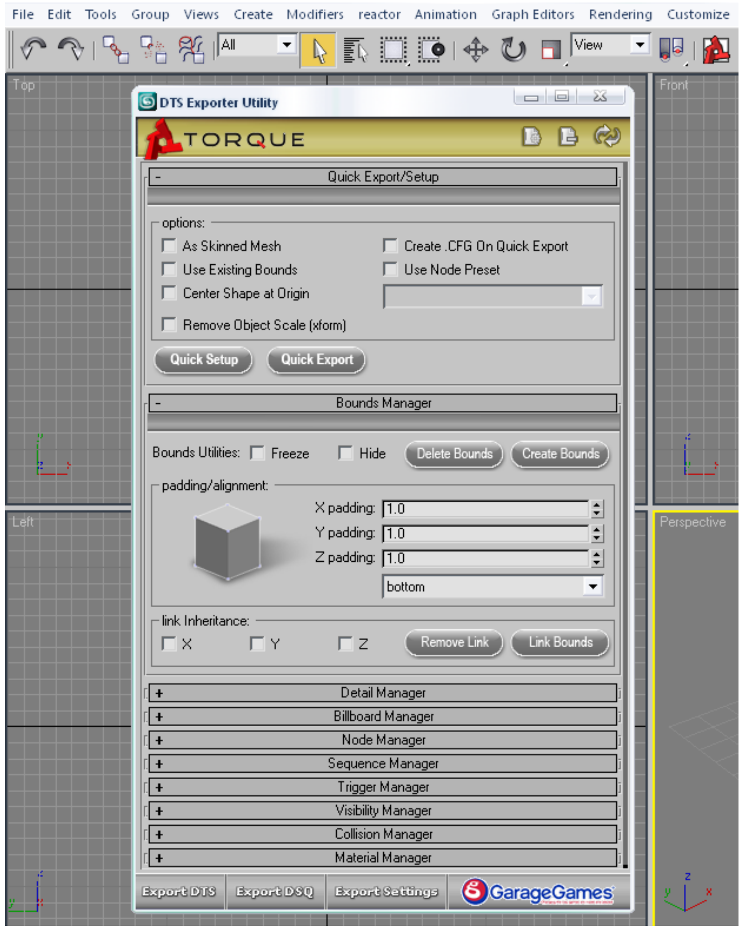


Figure 1-7 – TGE DTS Exporter Utility dialog box in 3DS Max

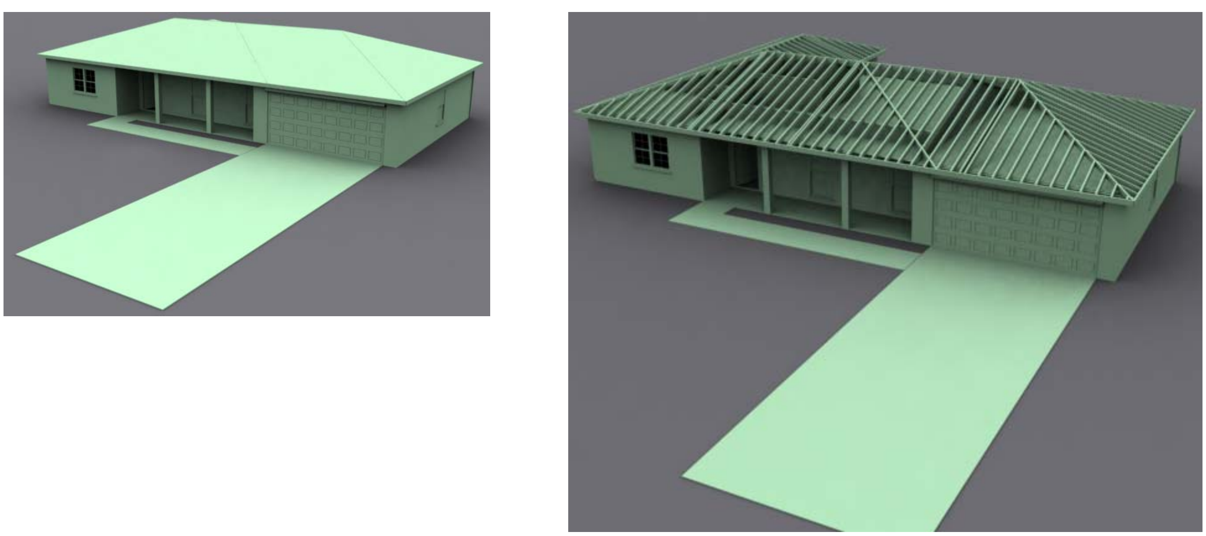


Figure 1-8 - Rendered images of the 3BR house using the Light Tracer rendering engine in 3DS Max

Step 2: Copying the 3D objects and textures into TGE folders

It is important to know where to place the 3D objects and their respective textures into the TGE folder structure. For this prototype design review application, the main folder structure is as follows:

```
VDRS [root folder]
        |--common          [TGE common engine folder]
        |--creator         [TGE Editor functions folder]
        |-- projectOne     [specific project level folder]
                |-- client
                |-- data
                |-- server
```

All converted 3D objects were placed under the data folder. In the data folder, sub-folders were created for easier management and identification purposes. The data sub-folder structure for the prototype application is as shown in Figure 1-9. Each type of 3D object was given its own folder and the same data folder structure will be shown in the prototype design review application.

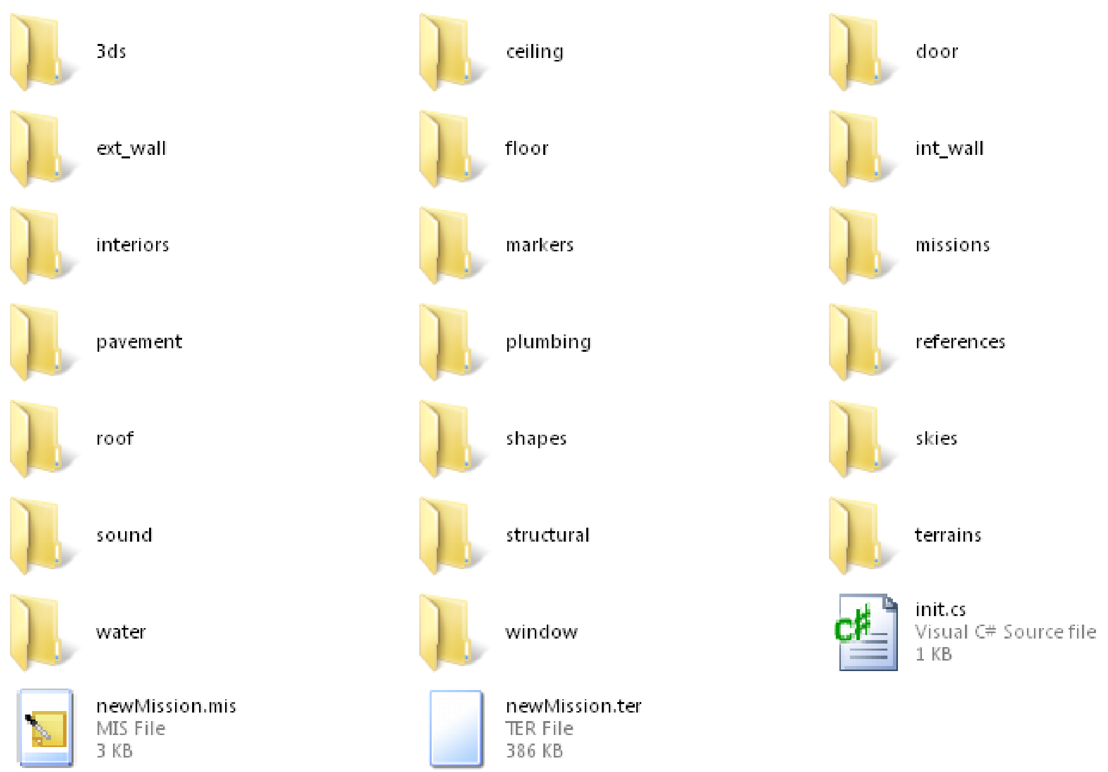


Figure 1-9 - Data sub-folder structure

Step 3: VE Environment Scene Assembling

As abovementioned, the data structure created in Step 2 was automatically recognized by the TGE and was correctly displayed in the prototype design review application World Editor (Figure 5-10).

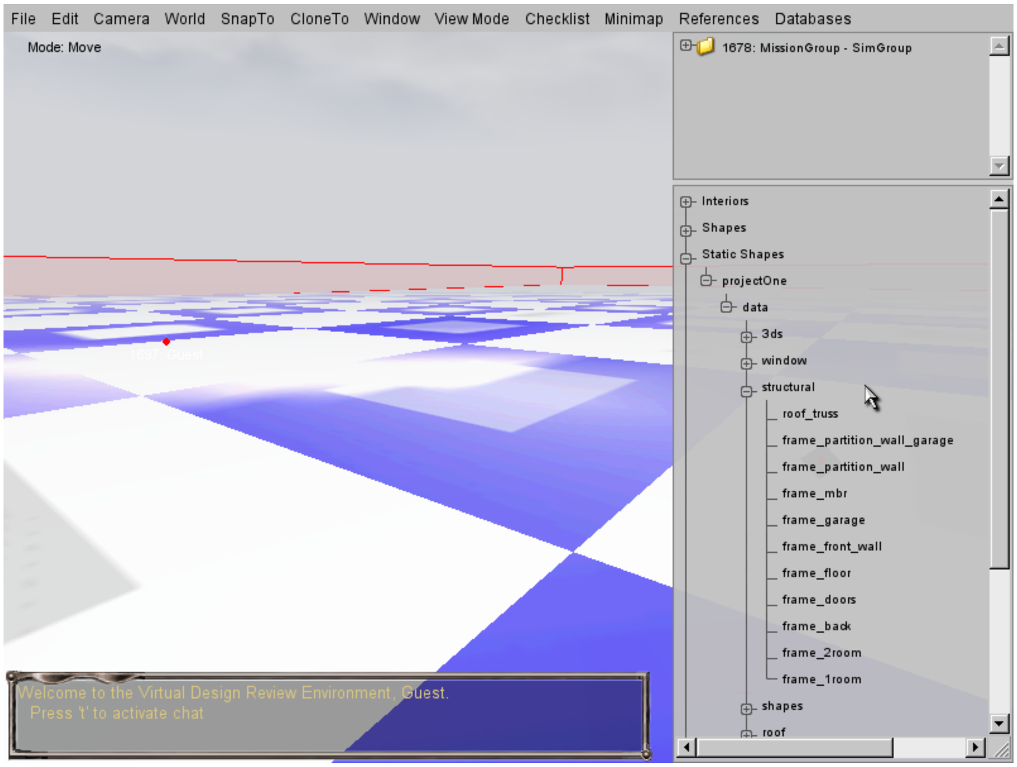


Figure 5-10 - Data sub-folder structure in the prototype application

The TGE was launched by double-clicking the main application executable file. Since the VE scene assembly can only be done in the World Editor Creator mode (Figure 1-11), the F11function key was used to switch to the World Editor mode. Next, by pressing the F4 function key, the World Editor Creator mode was invoked (or Main Menu → Window → World Editor Creator).

In the World Editor Creator mode, 3D objects can be placed, arranged and aligned in the VE.

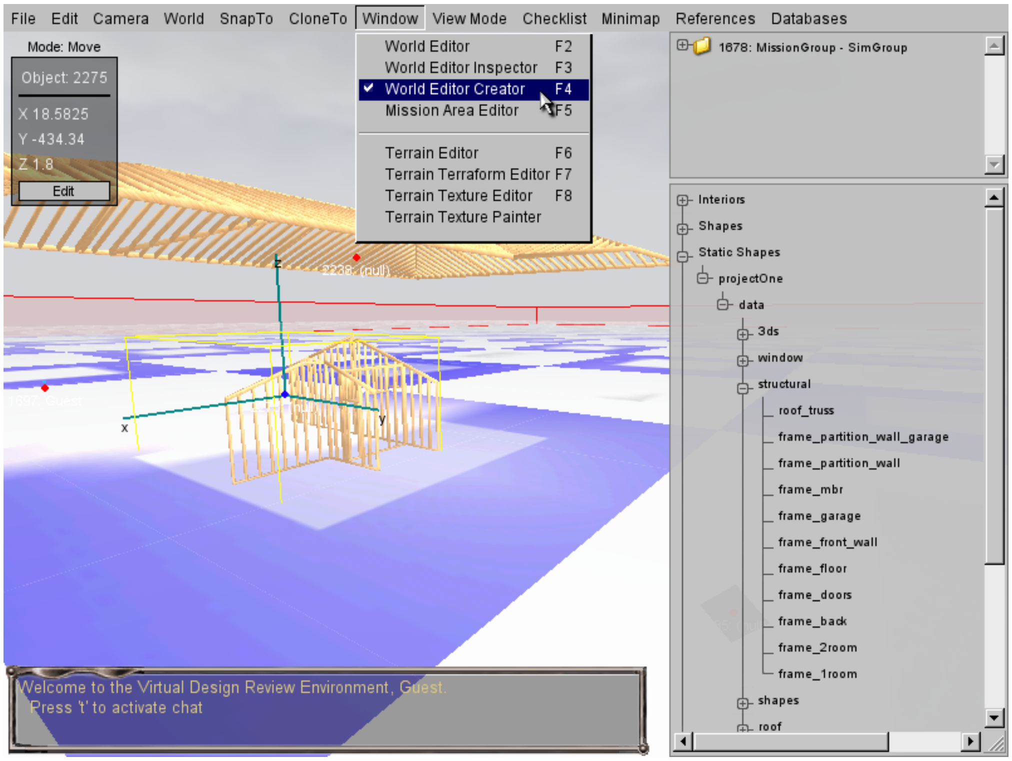


Figure 1-11 - Launching the World Editor Creator F4 for the VE scene building

One of the drawbacks of the TGE is it does not save the 3D objects coordinates from 3DS Max. This means that during the VE scene assembling process, all the 3D objects must be reassembled, placed and aligned properly. Besides programming the extra functionalities for the prototype design review application, reassembling the 3BR house model in the VE was one of the time-consuming processes. Different parts of the house were placed in the VE by selecting it in the 3D objects library. The 3D object was manually moved and aligned to match with other corresponding 3D objects in the VE. This process continued until the whole house was reassembled (Figure 1-12).

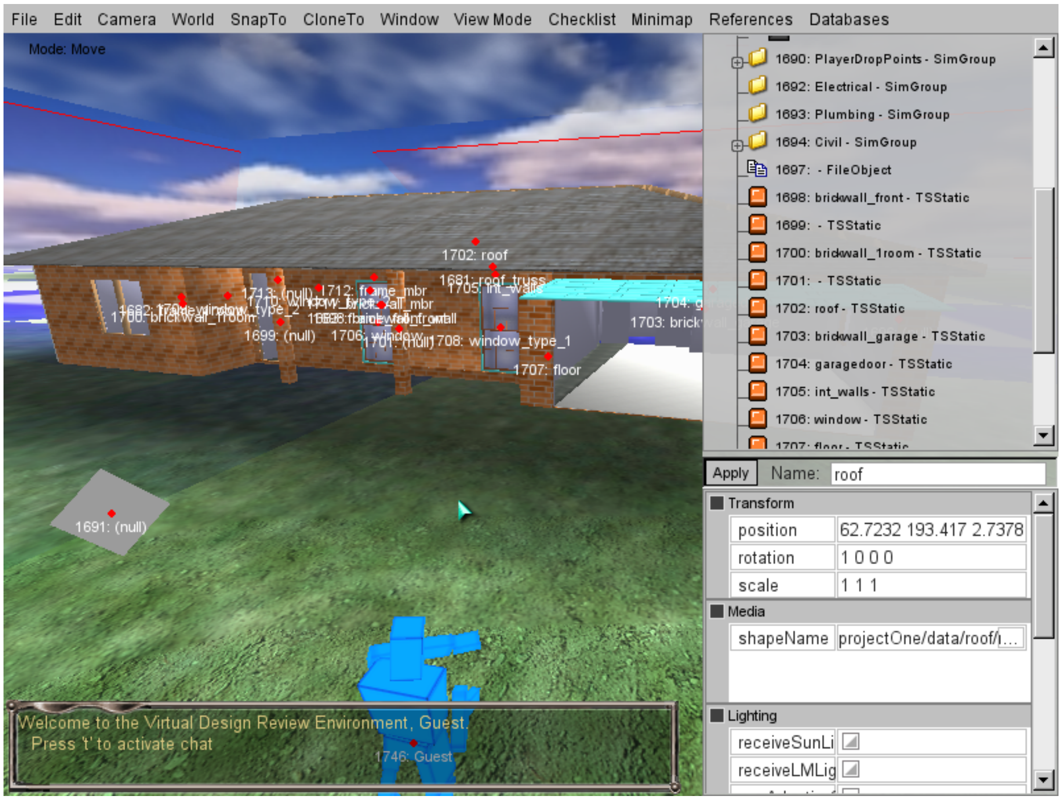


Figure 1-12 - The completely assembled 3D model of the 3BR house

In easing out the manual realignment process and to ensure precision, a new feature was developed to allow parametric inputs through an on-screen GUI. User has the ability to key in the three-dimensional X-Y-Z transformation coordinates of an object. This feature was crucial as it allows for a more precise placement and alignment of 3D objects in the VE. The GUI was developed in such a way that three different transformation modes i.e. Move, Rotate and Scale, can be executed using the same on-screen GUI. These three transformation modes can be accessed using the right-click menu (Figure 1-13). Details on the right-click menu are discussed in section 1.3.2.2.

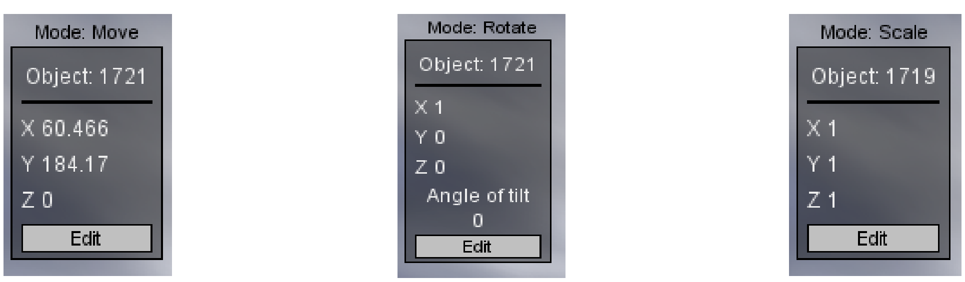


Figure 1-13 - On-screen GUI to allow for Parametric Inputs for the Move, Scale and Rotate transformation mode

In the VE, only sunlight was used instead of other types of lightings. Future implementation can include different types of lighting conditions.

5.3.2 Programming the Features of the Prototype Application

Two types of programming were performed: (1) introducing totally new codes to extend the functionality of TGE to support the prototype design review application, and (2) modifying the existing code structure to suit the prototype design review application features.

Due to the GaragaGames’ End-User-License-Agreement restrictions, original codes from GarageGames are not shown in this dissertation. However, code structures introduced by the author are included. Diagrams showing how the main features function are also included.

5.3.2.1 Design of the Graphical User Interface (GUI) for the Prototype

The Graphical User Interface (GUI) is a combination of the graphics and the scripts that carries the visual manifestation of the prototype design review application. The GUI then accepts the user's control inputs such as mouse movement or keyboard presses.

In the prototype design review application, part of the GUI elements includes:

1) the player's Heads Up Display (HUD)
2) the main start-up login screen window
3) the settings or option menus
4) the dialog boxes
5) the various in-world messaging systems such as text chat
6) right click menu
7) the hierarchical structure of the scene of the VE in the World Editor
8) pull down menus

Many of the GUI scripts in the TGE share the same basic script layout and properties. Depending on the type of GUI control, some may or may not require certain properties and therefore lines are added or omitted from the scripts themselves.

One of the useful development features of the TGE is the visual GUI designer. The F10 function key invoked the TGE GUI Designer (Figure 1-14). Using the TGE, GUI design for the prototype design review application was mainly developed visually by using the GUI Designer. Many of the GUI controls were already provided with some basic embedded properties. Further customizations were performed by keying in specific parameters in each given property box. Once a particular GUI design was developed, the TGE GUI Designer generated a custom GUI scripts for that particular GUI. In this dissertation, numerous GUI customizations were done to support the prototype design review application.

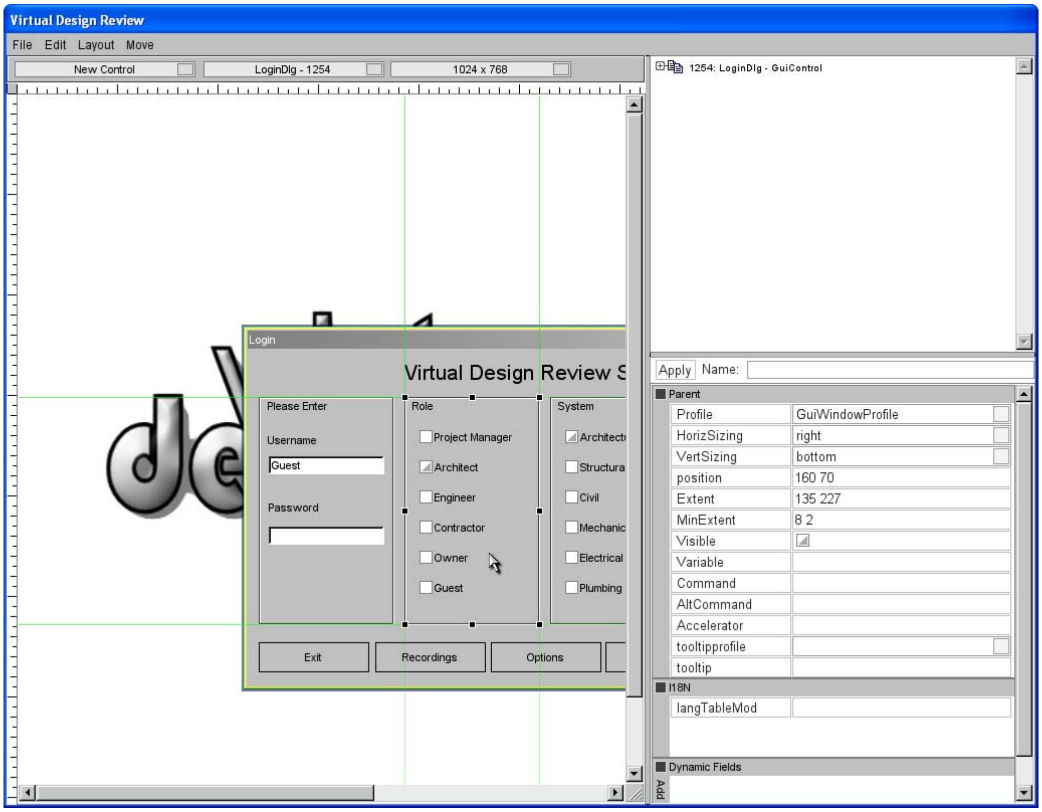


Figure 1-14 - Example screenshot of the TGE GUI Designer window

The main login screen is the first screen reviewers will see and it is the one of the most important features. Many of the parameters keyed in or selected by the reviewer started the filtering of the information (refer to Chapter 4 for further information on the context-aware processing of design review information).

Since the default state of TGE has no login system in place, new codes have to be developed. The login system allows a reviewer to host a design review session or join an existing session. It also allows a reviewer to choose a specific role during a design review session and the building systems the reviewer wishes to review.

Based on the suggestion given to the author during the interview with the local architect, the login system was implemented. Care was taken with the design of GUI, the information to be displayed, stored and retrieved from the login screen, the role of reviewer, and the level of security to be released to the reviewer. Because of the complexity of TGE security system, a password-protected system was not fully implemented. Ideally, a password-protected system should be put in place with some form of level of authority and access.

Figure 1-15 shows the flow diagram of the login system of the prototype design review application. A user enters a username and password, then selects a role and, chooses which building systems to review. The color of the Avatar will change based on the role selected. On the Main Login screen, the user can select whether to host a session or join as a client, edit the display options, view recording options, or exit the application.

If a user chooses to host a session, the user must select a project, check the “Host Design Review Session” option, and then click on the “Launch Session” button. Any password from the Main Login screen will be attached to the session. Other users who wish to join in must use the same password to log in. The “Host Design Review Session” option controls the availability/accessibility of the session to other users over a LAN or internet connection.

If the user chooses to join an existing design review session, the user must first locate a hosted session by hitting either the “Query LAN” or “Query Master” button. A search command will be issued to search for any existing session on the LAN and the internet respectively. To receive the most up-to-date information on a server that is hosting a session, the user can highlight it and use the “Refresh Server” button. Joining a hosted session is as simple as selecting it and hitting the “Join Server” button. As abovementioned, if a password exists on the server, it must match the password input on the Main Login screen.

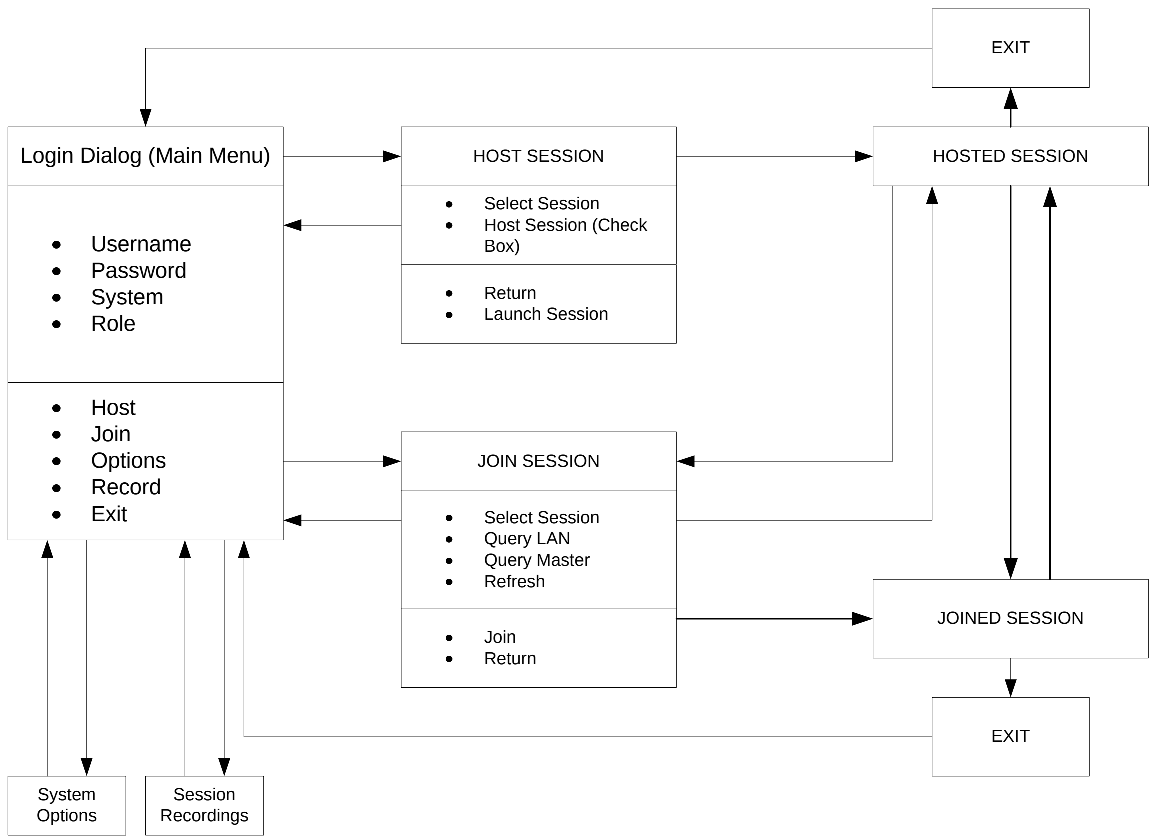


Figure 1-15 - Flow diagram of the login system

Using the “Select Role” section (Figure 1-16) from the main login screen, a description of a generic GUI script layout is shown below in Table 1-3.

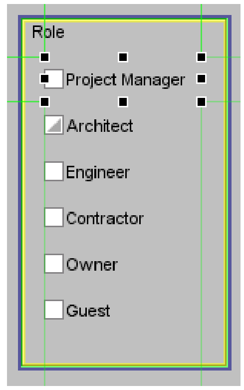


Figure 1-16 - The Select Role section from the main login screen

Line of script	**Description**
new GuiCheckBoxCtrl(a){	type of GUI control
Profile = "GuiCheckBoxProfile";	type of GUI profile
HorizSizing = "right";	horizontal size or length
VertSizing = "bottom";	vertical size or height
position = "15 25";	position of GUI on screen
Extent = "103 30";	
MinExtent = "8 2";	
Visible = "1";	GUI is set to be visible
Command = "LoginDlg.clearRole(); a.setValue(1); $PlayerChoice = \"green\"; LoginDlg.getModel();";	the command to be executed once user input is received
text = "Project Manager";	a text box that contains text string
groupNum = "-1";	
buttonType = "ToggleButton";	type of button used
};	end of function

Table 1-3 - Generic description of the lines of GUI scripts

As shown in the main login screen GUI of the prototype design review application, the GUI elements that exist on the login screen window alone are linked to other GUI scripts. The linkage between the scripts provides the sequence of events that will be executed based on the reviewer's input. E.g. referring to main start-up login screen GUI, when a reviewer hit the Join Screen button, the command "canvas.setContent(JoinServerGui);" was executed and opened-up the JoinServerGui window (Figure 1-17).

Figure **1-18** shows the linkages of all the GUI elements in the main login screen. Figure 1-19 shows some of the parameters available for customization of a GUI element.

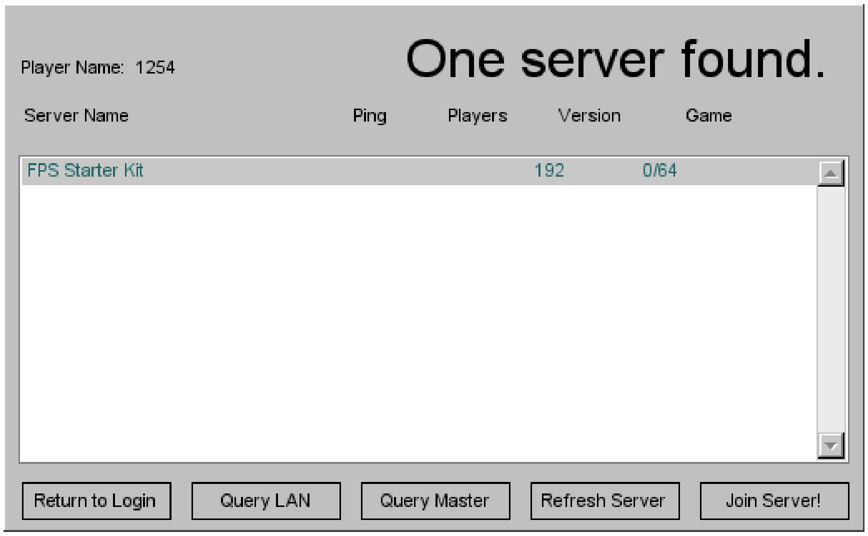


Figure 1-17 - The join server window

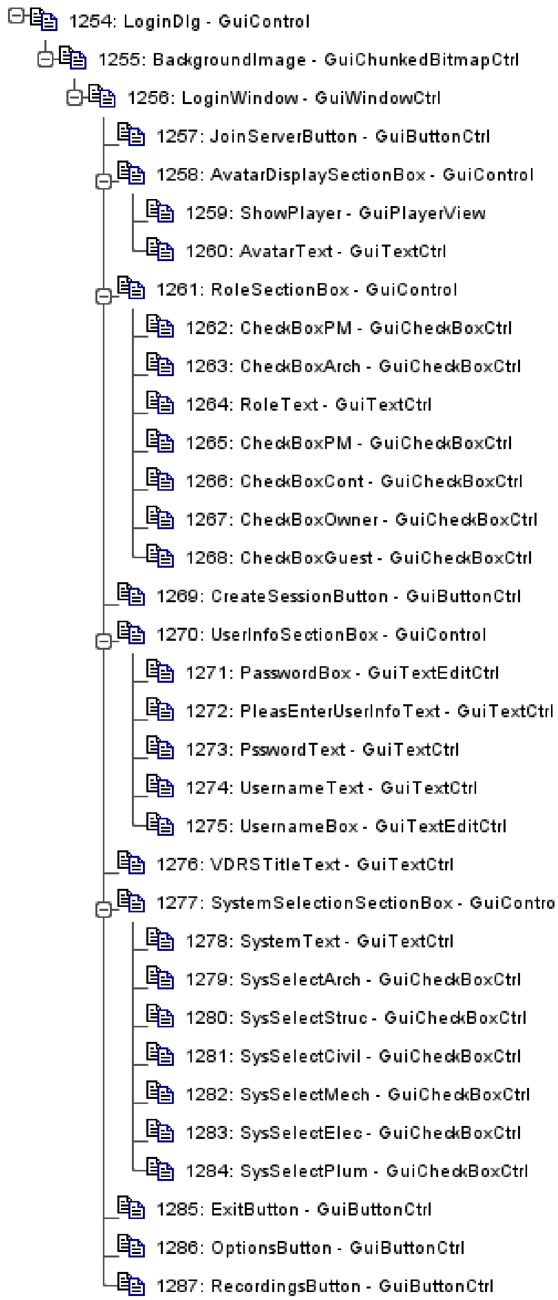


Figure 1-18 - The linkages of all the GUI elements in the main login screen.

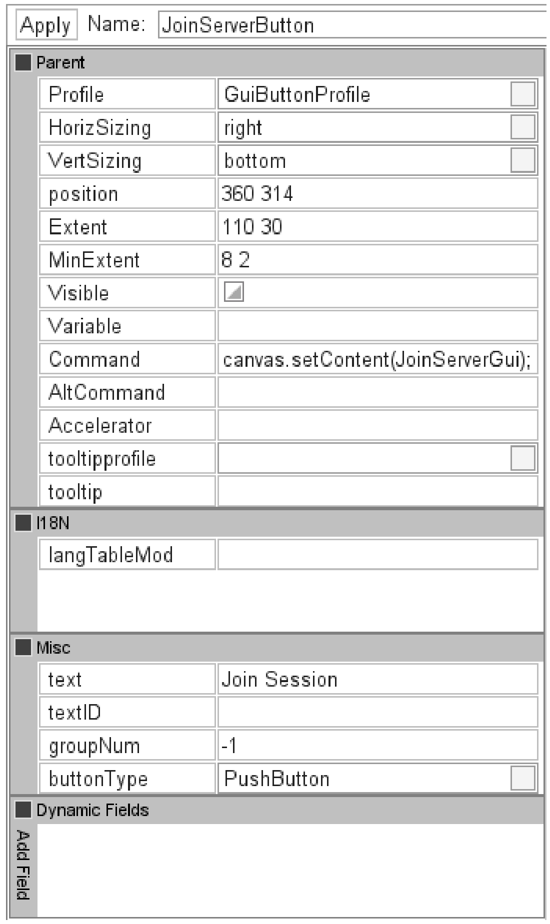


Figure 1-19 - Parameters available for further customization of a GUI element

Many of the functions from joinServerGui.gui and startMissionGui.gui were moved to the LoginDlg.cs file. The LoginDlg.cs worked together with the LoginDlg.gui. The LoginDlg.cs file can be found the /projectOne/client/ folder. Detailed LoginDlg.cs (Main Login screen) script code developed is included in Appendix 6.

Besides developing the function script for the login system, a GUI script was developed as well. The main function of this GUI script is to display any visual GUI element when needed or called by the function script. The rest of the LoginDlg.gui script is included in Appendix 7 and it is the one used for the main start-up login screen GUI of the prototype design review application. The LoginDlg.gui file can found under the /projectOne/client/ui/ folder. Each function was given a unique identity such as LoginWindow, JoinServerButton, ShowPlayer etc.

Once the script was tested and confirmed functional, it was then saved. Figure 1-20 shows the final GUI design of the main login screen.

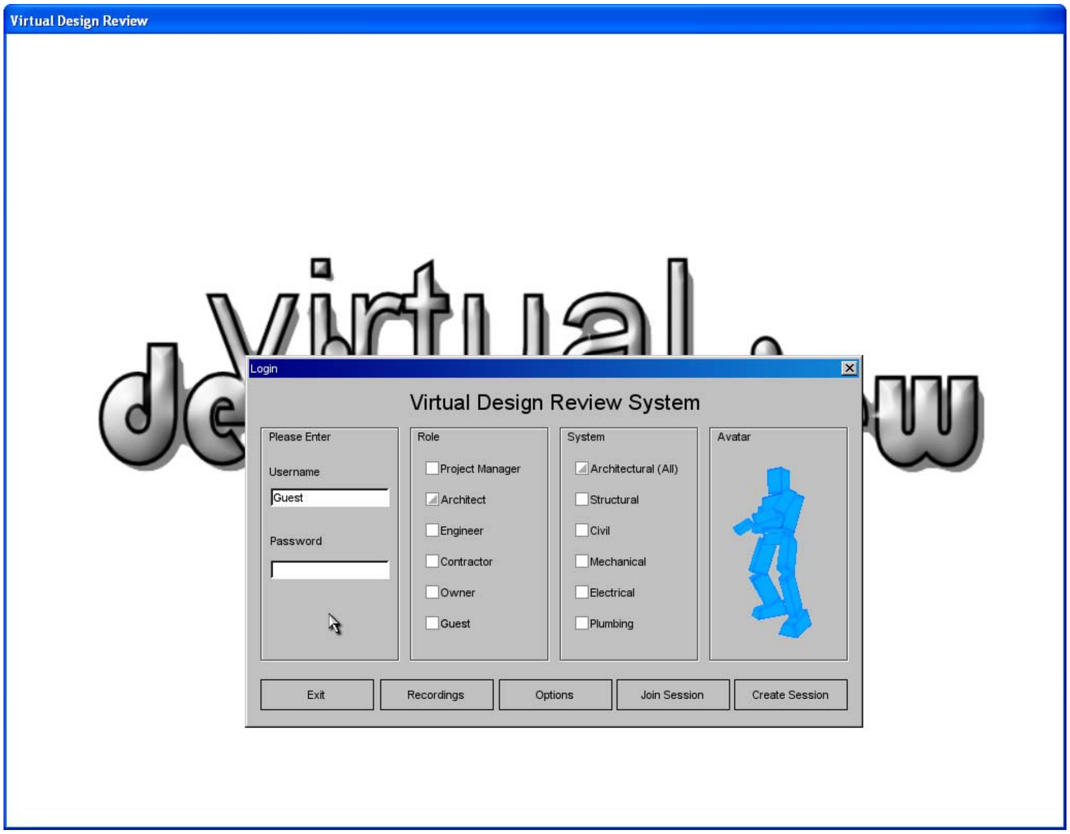


Figure 1-20 - The main login screen of the prototype design review application

5.3.2.2 3D Object Manipulation and the Right Click Menu System

From the discussion with the local architect, it was recommended that implementing a right click menu system was beneficial because the default implementation of TGE's World Editor was not intuitive enough. Many of the commands were either available only through the main menu, shortcut keys or a combination of both. The local architect was a proficient 2D and 3D CAD user and thus familiar to commands to be easily accessible through right-click menu system. However, for the prototype design review application, cascading style (or a second level menu) right click menu was not possible with the current available graphics functions in TGE. TGE graphics functions were limited to only 1-level menu system implementation. Adding this functionality may still be possible, but TGE OpenGL functions have to rewritten.

Having only 1-level menu system has limited the number of commands that can be made available on the right click menu. Once a draft right click menu layout was implemented, the author consulted the architect again to clarify which commands were regarded important during a design review. The final implementation of the right click menu is as shown in Figure 1-21. As abovementioned, the right-click menu was another feature that was suggested by the architect and added to the prototype application. Besides the parametric input for the object transformation, seven 3D object manipulation features were developed to allow for easier assembly of the VE. These 3D object manipulation features can be access through the right-click menu.

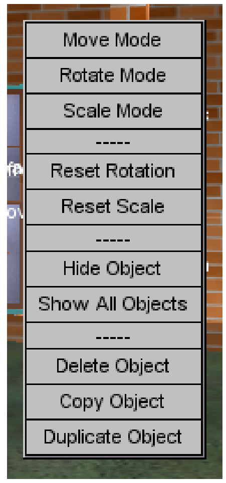


Figure 1-21 -The right-click menu

The function of each of the 3D object manipulation features introduced in the prototype design review application is shown in Table 1-4.

3D Object manipulation features	**Function**
Reset Rotation	Reset the rotation of the 3D object into its original state should there be any change in rotation made by the user
Reset Scale	Reset the scale of the 3D object into its original state should there be any change in scale made by the user
Hide Object	Hide a selected object or selection of objects
Show All Objects	Show all previously hidden objects
Delete Object	Delete an object
Copy Object	Make a copy of the master object and allow for pasting that copy in the VE. The copied object does not inherit properties of the master object (which means that characteristics of the copied object can be change and it is independent of the characteristics of the master object).
Duplicate Object	Make an exact duplicate of the master object. Duplicated object will inherit any changes made to the master object. E.g. several doors were duplicated from a master door. Any changes made to master door will be reflected onto duplicated doors. This feature allows for mass property change for similar 3D objects.

Table 1-4 - Description of the commands available through the right-click menu system

Note that the right-click menu was not only a collection of GUIs, but it also houses the various transformation commands that were part of real-time 3D object manipulation in the VE.

The inner-workings of the real-time 3D object manipulation and the right-click menu system is as follows. A popup menu will appear when an object is right clicked. It will disappear when an item in the menu is clicked or when the user clicked elsewhere in the VE besides the popup menu.

When right clicking occurred, a collision test will be performed by the engine. If an object besides the terrain, sky or water has been clicked, the selection will automatically be adjusted to include only that one particular object. A menu will then appear beginning from the center of the screen, which is the position the mouse is automatically transferred to the Object Manager after a right click is performed. The menu contains actions to perform on the single selected object as well as buttons to change the current 3D transformation mode.

The menu will disappear when an action from the menu is selected, or when any other portion of the VE is clicked. The popup menu is a window comprised of buttons with each of button corresponds to a different set of script functions. Each of these functions will be described in the next section.

The changes or insertion in codes to support these features are shown in Appendix 8.

5.3.2.3 Viewing Modes

In supporting the nature of both local and collaborative design review in a VE, various viewing modes were developed. These included the FPV (First-Person-View), TPV (Third-Person-View), BEV (Bird-Eye View), OV (Orbital View) and BSV (Behind-Shoulder View). A FPV allows design reviewer to view objects in the VE from the avatar’s eye. A TPV allows reviewer to view objects in the VE from a camera view just above and behind the avatar. A BEV takes the reviewer to view from a distance above looking down at the VE. An OV allows reviewers to circularly view the VE as the center object.

BSV is a new viewing mode introduced by the author to accommodate collaborative design review among reviewers in a VE. A BSV allows one reviewer to quickly shift the viewing FOV (field of view) to the location of another reviewer and have a view just behind that reviewer’s shoulder. This viewing mode allows for a reviewer to view what another reviewer is doing in the VE. The concept of BSV was originally thought by the author since text-chatting and to manually travel to the location of another reviewer was cumbersome. The implementation was demonstrated to the architect and it was agreed that BSV was a good viewing option especially during a collaborative design review session. The Tab key was programmed to accommodate the BSV.

5.3.2.4 Collaborative Design Review

The prototype design review application allows for real-time 3D object manipulation not only on a local PC, but also across the network with PCs connected either through the LAN or the

Internet. In a collaborative design review session, all forms of data are tracked on all client PCs that are connected to a Hosting Server. A client can now join a host PC over the internet or a LAN connection and perform 3D manipulation tasks in an almost identical fashion to that of the host. 3D manipulation tasks include moving, rotating and scaling objects, creating objects, deleting objects, and copying and pasting objects. These tasks can be executed simultaneously in real-time on the host as well as other client PCs.

The collaborative design review feature is one of the important contributions of this dissertation. The author has achieved to prove that a real-time collaborative VE can be developed using a low-cost 3D Game Engine; that allows for not only complex 3D object manipulation in real-time across the network, but also for real-time VE scene building, text chatting, information processing and real-time camera tracking among users. The author has successfully tested the collaborative design review feature with seven client PCs connected to a Hosting Server.

First numerous changes were made to TGE codes so the client is able to enter the World Editor mode (or Review mode) and load the scene just like the host is able to. Once the client is in the World Editor mode, custom codes had to be written to translate each command the client performs on the VE back to the server. Each manipulation in the VE results in a server command message being sent to the host with details on which objects are being manipulated and the new condition of the object. The server receives these server command messages and executes them in the order they are received. To avoid message overloads which can affect the real-time element from the VE, the number of messages the client sends to the host per second are adjusted accordingly. Once the host has executed an update message, the object in reference is updated back out to the clients just like it would have been with a traditional TGE scene. The host can now adjust how many times within a second these updates are sent out to the clients to help preserve the bandwidth if many clients are editing the level simultaneously and in real-time. Figure 1-22 shows a generic representation of the inner-workings of the messaging system of prototype design review application.

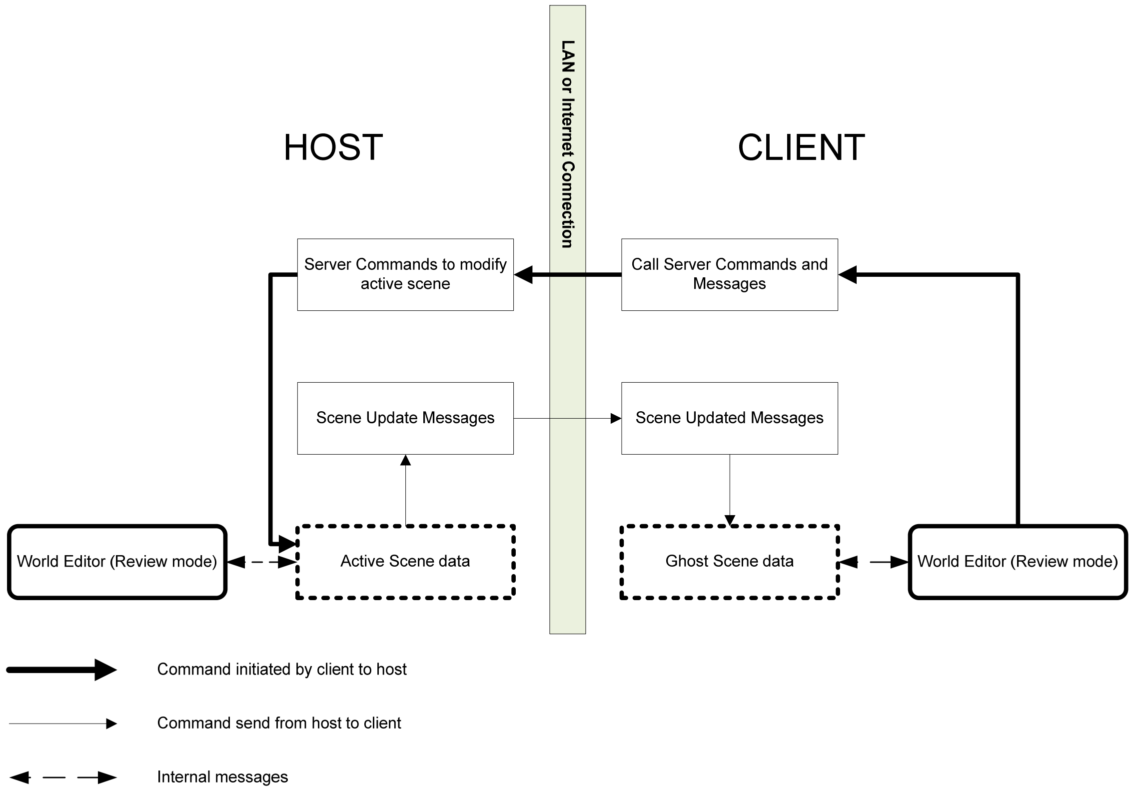


Figure 1-22 - The inner-workings of the messaging system of prototype design review application

For example, if Client A moves a brick column, an excerpt of the code within the worldEditor.cc executes a script function called *MoveObject*. Within the *MoveObject* function, the id number (the server will use to identify it; also known as ghost id) and the new position of the brick column are sent to the server using the *serverCmdMoveObject* server command. Many of these commands will be called in approximately one and a half second Client A takes to move the brick column. No intermediate messages will be culled by the client because Client A is moving that particular brick column, therefore avoiding clashes among clients. The host will receive these messages and execute each one in turn. The host will translate the ghost id received by Client A to determine which object Client A is moving and apply the new position. The new position is then updated to all other clients connected to the host. The result is a smooth moving brick column from the point of origin to final location specified by Client A, and this can be seen in real-time by the host and other clients (Figure 1-23).

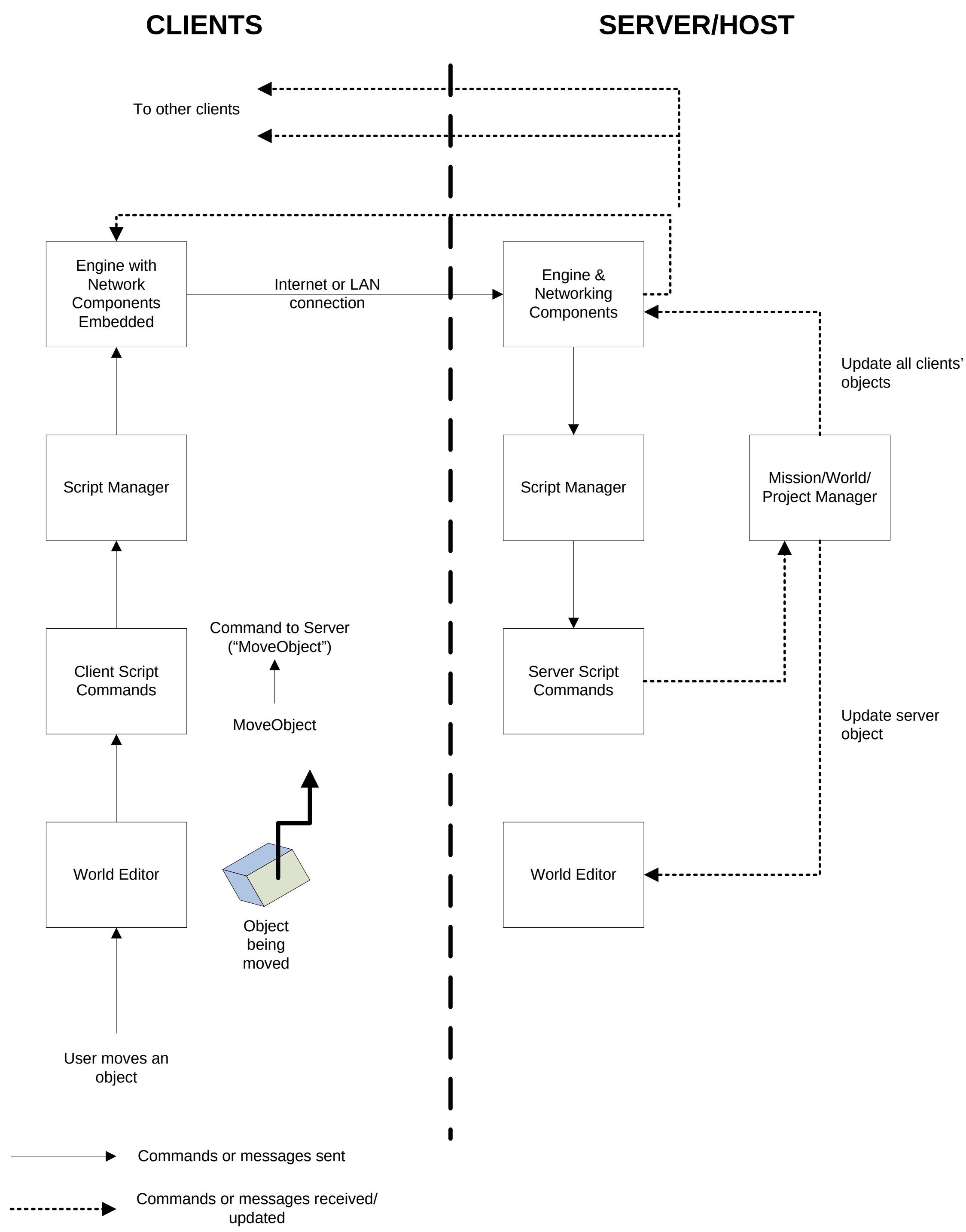


Figure 1-23 - Example of a client moving a 3D object during a collaborative design review session in the VE

5.3.2.5 Database Support for the Prototype Design Review Application

In supporting design review information processing, the author decided to use an open-source database software which is known as SQLite. It is the most widely used SQL database engine in the world. SQLite is different from SQL in that it offers no built-in server commands. This makes SQLite ideal for applications which already have a networking infrastructure, like the TGE, or that require no internet access such as cell phones or mp3 players. SQLite is specifically designed to be added to the source code of a software application to give the application a self-contained, persistent database system which requires no configuration (http://www.sqlite.org).

As mentioned earlier, the TGE is a low cost and flexible 3D game engine. However, there are still limitations when it comes to developing real-world and a more serious application. One of the proposed component for a design review application in a VE is information processing that includes data and information storage and retrieval. SQLite is the answer to this problem. For example, the easiest way to create persistent editable data with the TGE would be to use a text file format (such as comma-separated values: CSV and tab delimited text: TDT). CSV and TDT text file format can be opened and edited using most modern spreadsheet software application such as Microsoft Excel. However, using such file format has many problems. First, the data would be readable and editable using any text-capable reading software such a Notepad or WordPad. Second, managing the data would become increasingly time-consuming and difficult as the amount of data becomes larger. It may be easier to read in, processed and then sent out a few lines of information but for larger tasks, such as storing design review information, a proper database system is needed. In the prototype design review application, a rule-based approach was implemented using a text file format i.e. the CSV format. CSV format is used because it is easier to change or add items in the design review checklist, and there was no large data storage requirement. Whoever is responsible for the overall project's design would be able to modify design review checklists that can later be used by design reviewers.

Compared to other types of integration of database engine, SQLite is the easiest to implement into the TGE. Once SQLite was integrated into the prototype design review application, it creates a quick reference, encrypted database system which allows for many commands such as creating, deleting, querying, editing, and reorganizing the data. Some of the uses for SQLite that were implemented in the prototype design review application are logging the design reviewer's login information; store, retrieve and display design review comments and building codes; tracking object movements in the editor and embedded information such as dimensions, material; and cost of the objects in the VE.

The process of integrating SQLite into the TGE has already been achieved and documented by John Vanderbeck in 2004 (http://www.garagegames.com/index.php?sec=mg&mod=resource&page=view&qid=5531). A more detailed explanation of the implementation is described in the next section.

Figure 1-24 shows how information is bi-directionally passed between the user and the databases. Information is processed based on the user's request. The information is passed from the user through the Torque scripting code, and then through the SQLite code which is embedded in the TGE. From the TGE, the information is stored into the databases as persistent data. A *persistent data* is a term used to describe any data that will need to be written (stored) to a disk (storage device) and later can be retrieved and read back for further processing. In the prototype design review application, design review information is being passed between the user and the application.

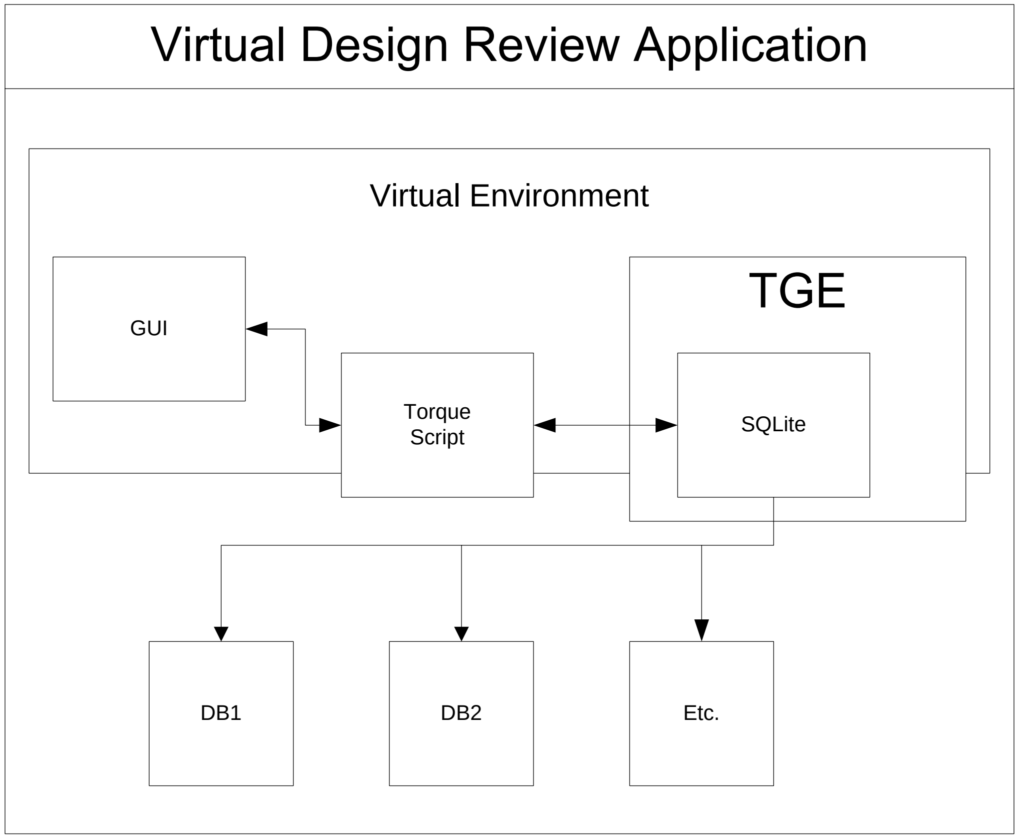


Figure 1-24 - Data flow diagram for the information passing through one component to the other

In the prototype design review application, the user often inputs the data through the GUI system. GUI implementations were in the forms of editable text boxes or multi-line editable text boxes. This information is then passed to the SQLite engine code in a string format using scripting code. Below is an example of a line of scripting code which might be used to make a query of a database. The '@' symbol represents the concatenation operation (connected or linked in a series) while the '%' before a word represents a local variable.

```
%res = %obj.query("SELECT * FROM data WHERE id= " @ %id @ " AND age<7 AND
name LIKE " @ %name);
```

The script handles all the details of each command so a user does not have to understand all the jargon needed to communicate with the SQLite engine. This feature makes the GUI design can be as simple as point and click, providing the GUI is set up to execute such commands. Once the GUI system executes the string commands, the string data is interpreted by the SQLite engine and then acts on the designated database according to the interpretation it derives from the string data it received.

Referring to the resources available on GarageGames' website, three steps were taken to embed the SQLite database into the prototype design review application.

1) The SQLiteObject.cc, SQLiteObject.h and sqlite.lib files were copied into the solution folder and added to the Visual Studio 2005 IDE compiler solution. Then the prototype application has to be recompiled.

2) The SQLite.dll was added to the folder that holds the prototype design review application executable. This step is important for the SQLite engine to properly run within the application. Without this, the application will produce error messages and crashes.

3) The sqlite.cs file was added from the resource to the list of script files used within the application. This allows for the use of simple Torque script commands to execute actions rather than having to know the complicated strings to pass to the SQLite engine (Figure 1-24). These functions can be executed as per application requirement. However, a GUI interface reduces the complexity and will not permit commands which may be dangerous and counterproductive such as deleting entire databases.

With the three steps described above, SQLite database support was embedded into the prototype design review application. A minor issue the author encountered was the issue of populating the databases with the necessary information. This has proven to be too time-consuming as no open-source GUI driven application exist that can open a SQL database. Unlike Microsoft Access which is GUI driven for easier population of data, most SQL database application has to be developed (standalone or web-based) or using third-party commercial software. The standalone implementation by the author supports direct input from within the prototype design review application into the SQLite databases. This form of implementation is crude and time-consuming. It is highly recommended that a third-party SQL commercial database editing software is used to speed up data entry for sizable databases. The scripts implemented for the prototype design review application is shown in Appendix 10.

The example shown below describes how a design reviewer would enter a set of specifications to an existing 3D object that does not have one.

In the VE, when a 3D object is selected, a query is automatically run on the object. The query is based on the object model’s name (object-centric mode). The query is run to determine if any specifications have been previously assigned to the object. If the specifications do exist, a pop-up window will appear to allow the design reviewer to click and view the specifications dialog window. If no specifications have been defined, the same popup window will appear. The design reviewer can click on the “Create Specifications’ button. An empty “Edit Specifications” window will appear and the design reviewer can enter the appropriate data for the 3D object. Once completed, the design reviewer can click the “Save and Return” button (Figure 1-25).

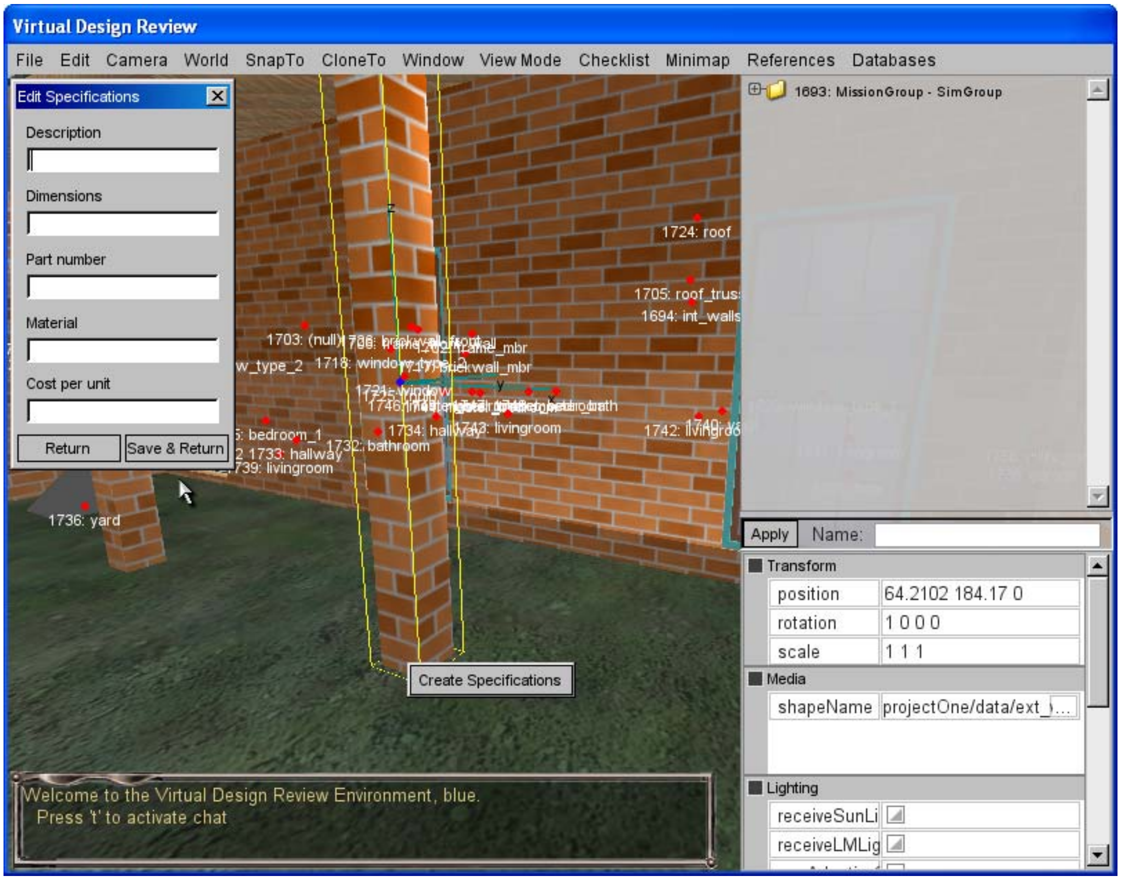


Figure 1-25 - The Edit Specification window to allow for addition of new information

This action assigns the data to the SQLite database as defined by the SpecsEditDlg::onSave function. Upon assigning the specification information to a 3D object, the dialog window which displays the specification data is automatically opened and displays the data of the object. The SpecsMenu::onWake function queries and returns the data from the SQLite database and then displays it to the user in the specifications window. All object specification data is stored in the SPECS SQLite database file in the same folder as the design review application. Objects are sorted first by model name (description) within the SPECS file. All other data such as cost, dimensions, material etc. are created and stored as secondary fields. The specification information can be edited at any time to correct any errors or to supply updated information.

5.3.3 Features Implemented in the Work-In-Progress (WIP) Prototype Design Review Application

A work-in-progress (WIP) prototype design review application was developed using the Torque 3D Game Engine (TGE). All the features developed are new, and none exist in the stock TGE (with the exception of FPS, TPS, Orbital and Bird's Eye view mode, the BSV is a new viewing feature). Some of the specific features implemented in the prototype design review application are shown in Table 1-5.

#	Feature	Description
1	Context-Aware	Location-aware – through embedded location sensors placed in the VE, the application is able detect changes of design reviewer's location. Discipline-aware – based on the response received when a design reviewer logs in, application is able to display specific view based on the selected discipline. Only architectural and structural discipline was implemented. Only these two context-aware were implemented in the WIP prototype application.
2	Database support	Dynamic databases created using SQLite is fully supported in the prototype design review application. Some of the databases implemented are logs and object information. A lessons-learned static database was also created using Microsoft Windows Help file format. This was just an option should developer take this route to create a database. Static database does not allow for real-time information processing.
3	Rule-based checklist	This was implemented using Comma-Separated Values file.
4	Real-time collaboration across network	Real-time collaboration across the network supports 3D object manipulation, text-chat and various viewing mode. To date, the author tested on seven client-computes accessing one host-computer. No significant lag was noticed.
5	View modes & Navigation	First-Person, Third-Person, Orbital, Birds-Eye, Behind-Shoulder were implemented to support collaborative design review. The Behind-Shoulder view is a new concept developed by the author. Navigations in the VE were closely tied to the viewing modes. Flythrough and Walkthrough in the VE were the two navigation modes implemented in the prototype design review application.
6	3D Object Manipulation	Objects can be placed, moved, scaled, rotated, and deleted in the VE.
7	Parametric input	Parametric input was implemented as an option to object-manipulation in the VE. Parametric input helps precise object movement, scaling and rotation.

8	GUI	Various GUIs including login screen and right-click menu were developed to support various design review tasks in the VE.
9	3D Object database and organization	A more organized 3D object directory file structure was introduced. The default hierarchical-tree support present in the TGE was improved. 3D object can be hide or shown through right-click.

Table 1-5 - Some of the features developed in the WIP prototype design review application

5.4 Conclusion

This appendix described the development process of the prototype design review application. This chapter also described the components and structure of the TGE. Only suitable and usable components of the TGE were used and incorporated into the prototype design review application. Two major development phases took place; development of VE, and modification of existing or the introduction of new functionalities through programming (such as the implementation of real-time 3D object manipulation, real-time collaboration, networking, GUI, information processing and visualization). The prototype design review application was developed in several stages, and each a time a milestone was achieved, the author consulted a local architect for his expert opinion. The components for the design review system were implemented in the prototype design review application. In conclusion, the author has achieved in developing a workable prototype design review application. The author has also defined the requirements for developing a design review application in a VE utilizing a 3D Game Engine.

Appendix 6: Codes for the LoginDlg.cs

```
// LoginDlg.cs
function LoginDlg::clearRole(%this) // Uncheck all "Role" boxes
{
   a.setValue(0);
   b.setValue(0);
   c.setValue(0);
   d.setValue(0);
   e.setValue(0);
   f.setValue(0);
}

function LoginDlg::clearSystem(%this) // Uncheck all "System" boxes
{
   j.setValue(0);
   k.setValue(0);
   l.setValue(0);
   m.setValue(0);
   n.setValue(0);
   o.setValue(0);
}

function LoginDlg::loginRec(%this)// Record who/when is logging in
{
   $name=username.getValue();
   %file = new FileObject();
   %t= getLocalTime(); // Credit to Drew Parker for writting the
getLocalTime() function
   %file.OpenForAppend("./Log/" @ username.getValue() @ ".txt");// get
username and save it as logifile username.txt
   %file.writeline("");
   %file.writeline("");
   %file.writeline("Username    -  " @ username.getValue());
   %file.writeline("  Login time  -  month:" @ getWord (%t, 0) @ ", day: " @
getWord (%t, 1) @ ", year: " @ getWord (%t, 2) @ ", hour: " @ getWord (%t, 3)
@ ", min: " @ getWord (%t, 4) @ ", sec: " @ getWord (%t, 5));
   (%file.getId()).delete();
   appendDatabase("LOG", "");
   appendDatabase("LOG", "");
   appendDatabase("LOG", "Username    -  " @ username.getValue());
if($IsClient)
      appendDatabase("LOG", " Joining an existing session...");
else
      appendDatabase("LOG", " Creating a new session...");
   appendDatabase("LOG", "  Login time  -  month:" @ getWord (%t, 0) @ ",
day: " @ getWord (%t, 1) @ ", year: " @ getWord (%t, 2) @ ", hour: " @
getWord (%t, 3) @ ", min: " @ getWord (%t, 4) @ ", sec: " @ getWord (%t, 5));
}

function LoginDlg::systemRec(%this, %system)// Record which building
subsystem is being reviewed
{
    %file = new FileObject();
```

```
    %t= getLocalTime(); //Credit to Drew Parker for writting the
getLocalTime() function
    %file.OpenForAppend("./Log/" @ username.getValue() @ ".txt");
    %file.writeline("   " @ %system @ " checklist reviewed -  day: " @
getWord (%t, 1) @ ", hour: " @ getWord (%t, 3));
    (%file.getId()).delete();
    appendDatabase("LOG", "   " @ %system @ " checklist reviewed -  day: " @
getWord (%t, 1) @ ", hour: " @ getWord (%t, 3));
}

function LoginDlg::manualStartRec(%this)
{
    %file = new FileObject();
    %t= getLocalTime(); //Credit to Drew Parker for writting the
getLocalTime() function
    %file.OpenForAppend("./Log/" @ username.getValue() @ ".txt");
    %file.writeline("           Manual Compliance Checklist started at -
day: " @ getWord (%t, 1) @ ", hour: " @ getWord (%t, 3));
    (%file.getId()).delete();
    appendDatabase("LOG", "           Manual Compliance Checklist started at
-  day: " @ getWord (%t, 1) @ ", hour: " @ getWord (%t, 3));
}

function LoginDlg::manualEndRec(%this)
{
    %file = new FileObject();
    %t= getLocalTime(); //Credit to Drew Parker for writting the
getLocalTime() function
    %file.OpenForAppend("./Log/" @ username.getValue() @ ".txt");
    %file.writeline("           Manual Compliance Checklist finished at -
day: " @ getWord (%t, 1) @ ", hour: " @ getWord (%t, 3));
    (%file.getId()).delete();
    appendDatabase("LOG", "           Manual Compliance Checklist finished at
-  day: " @ getWord (%t, 1) @ ", hour: " @ getWord (%t, 3));
}

function LoginDlg::commentRec(%this)
{
    %file = new FileObject();
    %t= getLocalTime(); //Credit to Drew Parker for writting the
getLocalTime() function
    %file.OpenForAppend("./Log/" @ username.getValue() @ ".txt");
    %file.writeline("     Comment Dialog Saved -  day: " @ getWord (%t, 1) @
", hour: " @ getWord (%t, 3) @ ", min: " @ getWord (%t, 4) @ ", sec: " @
getWord (%t, 5));
    (%file.getId()).delete();
}

function LoginDlg::responseRec(%this, %response)
{
    %file = new FileObject();
    %t= getLocalTime(); //Credit to Drew Parker for writting the
getLocalTime() function
    %file.OpenForAppend("./Log/" @ username.getValue() @ ".txt");
    %file.writeline("       Question " @ $checklistCount-1 @ " - " @
%response @ " -  day: " @ getWord (%t, 1) @ ", hour: " @ getWord (%t, 3) @ ",
min: " @ getWord (%t, 4) @ ", sec: " @ getWord (%t, 5));
```

```
    (%file.getId()).delete();
    appendDatabase("LOG", "       Question " @ $checklistCount-1 @ " - " @
%response @ " -  day: " @ getWord (%t, 1) @ ", hour: " @ getWord (%t, 3) @ ",
min: " @ getWord (%t, 4) @ ", sec: " @ getWord (%t, 5));
}

function LoginDlg::logoutRec(%this)
{
    %file = new FileObject();
    %t= getLocalTime(); //Credit to Drew Parker for writting the
getLocalTime() function
    %file.OpenForAppend("./Log/" @ username.getValue() @ ".txt");
    %file.writeline("  Logout time -  month:" @ getWord (%t, 0) @ ", day: " @
getWord (%t, 1) @ ", year: " @ getWord (%t, 2) @ ", hour: " @ getWord (%t, 3)
@ ", min: " @ getWord (%t, 4) @ ", sec: " @ getWord (%t, 5));
    (%file.getId()).delete();
    appendDatabase("LOG", "  Logout time -  month:" @ getWord (%t, 0) @ ",
day: " @ getWord (%t, 1) @ ", year: " @ getWord (%t, 2) @ ", hour: " @
getWord (%t, 3) @ ", min: " @ getWord (%t, 4) @ ", sec: " @ getWord (%t, 5));
}

function LoginDlg::getSystem(%this) // Displays which Building System has
been selected
{
if(j.getValue())
return("MissionGroup");
if(k.getValue())
return("Structural");
if(l.getValue())
return("Civil");
if(m.getValue())
return("Mechanical");
if(n.getValue())
return("Electrical");
if(o.getValue())
return("Plumbing");
}

function LoginDlg::onWake(%this)
{
 $PlayerChoice = "blue"; // by default the avatar color is blue
 LoginDlg.getModel(); // retrieve the avatar model
if(!(isFile("LOG")))
    CreateDatabase("LOG"); // create a logfile

if(!(isFile("Codes")))
    CreateDatabase("Codes");
if(!(isFile("LessonsLearned")))
    CreateDatabase("LessonsLearned");
if(!(isFile("Personnel")))
    CreateDatabase("Personnel");
if(!(isFile("Checklists")))
    CreateDatabase("Checklists");
if(!(isFile("Objects")))
    CreateDatabase("Objects");
if(!(isFile("2DDrawings")))
    CreateDatabase("2DDrawings");
```

```
if(!(isFile("Standards")))
    CreateDatabase("Standards");
if(!(isFile("BasisOfDesign")))
    CreateDatabase("BasisOfDesign");

 LoginDlg.clearRole();
 j.setValue(1); // Default values for check boxes
 b.setValue(1);
}

function LoginDlg::getModel(%this)
{
 ShowPlayer.setModel("ProjectOne/data/shapes/player/player_" @ $PlayerChoice
@ "/player.dts", "ProjectOne/data/shapes/player/" @ $PlayerChoice @
"player_forward.dsq");
 $pref::Player::Color = $PlayerChoice; // load player avatar model
}

function LoginDlg::onSleep(%this)
{
 $pref::Player::Name=username.getValue();
}

// Session Loading Screen

function SM_StartMission()
{
   %id = SM_missionList.getSelectedId();
   %mission = getField(SM_missionList.getRowTextById(%id), 1);

if ($pref::HostMultiPlayer)
      %serverType = "MultiPlayer";
else
      %serverType = "SinglePlayer";

   createServer(%serverType, %mission);
   %conn = new GameConnection(ServerConnection);
   RootGroup.add(ServerConnection);
   %conn.setConnectArgs($pref::Player::Name, $Pref::Player::Color);
   %conn.setJoinPassword($Client::Password);
   %conn.connectLocal();
}

function startMissionGui::onWake()
{
   initServer2();
   user2.setText($pref::Player::Name);
   SM_missionList.clear();
   %i = 0;
for(%file = findFirstFile($Server::MissionFileSpec); %file !$= ""; %file =
findNextFile($Server::MissionFileSpec))
if (strStr(%file, "/CVS/") == -1)
         SM_missionList.addRow(%i++, getMissionDisplayName(%file) @ "\t" @
%file );
   SM_missionList.sort(0);
   SM_missionList.setSelectedRow(0);
   SM_missionList.scrollVisible(0);
```

```
}


function getMissionDisplayName( %missionFile )
{
   %file = new FileObject();

   %MissionInfoObject = "";

if ( %file.openForRead( %missionFile ) ) {
            %inInfoBlock = false;

            while ( !%file.isEOF() ) {
                  %line = %file.readLine();
                  %line = trim( %line );

                  if( %line $= "new ScriptObject(MissionInfo) {" )
                        %inInfoBlock = true;
                  elseif( %inInfoBlock && %line $= "};" ) {
                        %inInfoBlock = false;
                        %MissionInfoObject = %MissionInfoObject @ %line;
                        break;
                  }

                  if( %inInfoBlock )
                     %MissionInfoObject = %MissionInfoObject @ %line @ " ";
            }

            %file.close();
      }
      %MissionInfoObject = "%MissionInfoObject = " @ %MissionInfoObject;
      eval( %MissionInfoObject );

   %file.delete();

if( %MissionInfoObject.name !$= "" )
return %MissionInfoObject.name;
else
return fileBase(%missionFile);
}


// Join Server Screen
//------------------------------------------
function JoinServerGui::onWake()
{
JS_joinServer.setActive(JS_serverList.rowCount() > 0);
   user.setText($pref::Player::Name);
   initClient2();
}


function JoinServerGui::query(%this)
{
   queryMasterServer(
      0,                          // Query flags
      $Client::GameTypeQuery,     // gameTypes
```

```
      $Client::MissionTypeQuery,    // missionType
      0,                            // minPlayers
      100,                          // maxPlayers
      0,                            // maxBots/npc
      2,                            // regionMask
      0,                            // maxPing
      100,                          // minCPU
      0                             // filterFlags
      );
}

function JoinServerGui::queryLan(%this)
{
   queryLANServers(
      28000,                        // lanPort for local queries
      0,                            // Query flags
      $Client::GameTypeQuery,       // gameTypes
      $Client::MissionTypeQuery,    // missionType
      0,                            // minPlayers
      100,                          // maxPlayers
      0,                            // maxBots
      2,                            // regionMask
      0,                            // maxPing
      100,                          // minCPU
      0                             // filterFlags
      );
}

function JoinServerGui::cancel(%this)
{
   cancelServerQuery();
   JS_queryStatus.setVisible(false);
}

function JoinServerGui::join(%this)
{
   cancelServerQuery();
   %id = JS_serverList.getSelectedId();

// The server info index is stored in the row along with the
// rest of displayed information
   %index = getField(JS_serverList.getRowTextById(%id),6);
if (setServerInfo(%index)) {
      %conn = new GameConnection(ServerConnection);
      %conn.setConnectArgs($pref::Player::Name, $Pref::Player::Color);
      %conn.setJoinPassword($Client::Password);
      %conn.connect($ServerInfo::Address);
   }
}
function JoinServerGui::refresh(%this)
{
   cancelServerQuery();
   %id = JS_serverList.getSelectedId();

// The server info index is stored in the row along with the
// rest of displayed information
   %index = getField(JS_serverList.getRowTextById(%id),6);
```

```
if (setServerInfo(%index)) {
      querySingleServer( $ServerInfo::Address, 0 );
   }
}

function JoinServerGui::refreshSelectedServer( %this )
{
   querySingleServer( $JoinGameAddress, 0 );
}

function JoinServerGui::exit(%this)
{
   cancelServerQuery();
   Canvas.setContent(mainMenuGui);
}

function JoinServerGui::update(%this)
{
// Copy the servers into the server list.
   JS_queryStatus.setVisible(false);
   JS_serverList.clear();
   %sc = getServerCount();
for (%i = 0; %i < %sc; %i++) {
      setServerInfo(%i);
      JS_serverList.addRow(%i,
         $ServerInfo::Name TAB
         $ServerInfo::Ping TAB
         $ServerInfo::PlayerCount @ "/" @ $ServerInfo::MaxPlayers TAB
         %i);  // ServerInfo index stored also
   }
   JS_serverList.sort(0);
   JS_serverList.setSelectedRow(0);
   JS_serverList.scrollVisible(0);

   JS_joinServer.setActive(JS_serverList.rowCount() > 0);
}

function onServerQueryStatus(%status, %msg, %value)
{
      echo("ServerQuery: " SPC %status SPC %msg SPC %value);
// Update query status
// States: start, update, ping, query, done
// value = % (0-1) done for ping and query states
if (!JS_queryStatus.isVisible())
      JS_queryStatus.setVisible(true);

switch$ (%status) {
case"start":
         JS_joinServer.setActive(false);
         JS_queryMaster.setActive(false);
         JS_statusText.setText(%msg);
         JS_statusBar.setValue(0);
         JS_serverList.clear();

case"ping":
         JS_statusText.setText("Ping Servers");
         JS_statusBar.setValue(%value);
```

```
case"query":
         JS_statusText.setText("Query Servers");
         JS_statusBar.setValue(%value);

case"done":
         JS_queryMaster.setActive(true);
         JS_queryStatus.setVisible(false);
         JS_status.setText(%msg);
         JoinServerGui.update();
   }
}

function initClient2()
{
   echo("\n--------- Joining Session as Client ---------");

   %filename = "projectOne/data/shapes/player/player_*/player2.cs";
for (%file = findFirstFile(%filename); %file !$= ""; %file =
findNextFile(%filename))
       exec(%file);

// Make sure this variable reflects the correct state
   $Server::Dedicated = false;

// Game information used to query the master server
   $Client::GameTypeQuery = "FPS Starter Kit";
   $Client::MissionTypeQuery = "Any";

   initBaseClient();
   setNetPort(0);
}

function initServer2()
{
   echo("\n--------- Initializing Session as Server ---------");

// Server::Status is returned in the Game Info Query and represents the
// current status of the server. This string should be very short.
   $Server::Status = "Unknown";

// Turn on testing/debug script functions
   $Server::TestCheats = false;

// Specify where the mission/session files are located/stored
   $Server::MissionFileSpec = "*/missions/*.mis";

}
```

Appendix 7: Codes for the LoginDlg.gui

```
// A new GUI element saved as LoginDlg.gui file and later on called by the
parent script during execution
//--- OBJECT WRITE BEGIN ---
newGuiControl(LoginDlg) {     // main function for the Login window
   Profile = "GuiDefaultProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "0 0";
   Extent = "1024 768";
MinExtent = "8 2";
   Visible = "1";
   Variable = "ame";
   Command = "ame";
AltCommand = "ame";
   Accelerator = "ame";
tooltip = "ame";
langTableMod = "ame";

// defines the Background image properties
newGuiChunkedBitmapCtrl(BackgroundImage) { // Background image on the Main
Menu
      Profile = "GuiDefaultProfile";
HorizSizing = "width";
VertSizing = "height";
position = "0 0";
      Extent = "1024 768"; // application default window size
MinExtent = "8 2";
      Visible = "1";
bitmap = "./background.jpg"; // background image filename
useVariable = "0";
tile = "0"; // image is not tiled

// defines the properties of the login window
newGuiWindowCtrl(LoginWindow) {
         Profile = "GuiWindowProfile"; // profile type = window
HorizSizing = "right";
VertSizing = "bottom";
position = "160 308";
         Extent = "599 361";
MinExtent = "8 2";
         Visible = "1";
text = "Login";
maxLength = "255";
resizeWidth = "0";
resizeHeight = "0";
canMove = "1";
canClose = "1";
canMinimize = "0";
canMaximize = "0";
minSize = "50 50";

// defines the properties of JoinServerButton
```

```
newGuiButtonCtrl(JoinServerButton) {
            Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "360 314";
            Extent = "110 30";
MinExtent = "8 2";
            Visible = "1";
            Command = "canvas.setContent(JoinServerGui);";
text = "Join Session";
groupNum = "-1";
buttonType = "PushButton";
         };

// defines the properties of Avatar Display Section Box
newGuiControl(AvatarDisplaySectionBox) {
            Profile = "GuiWindowProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "450 70";
            Extent = "135 227";
MinExtent = "8 2";
            Visible = "1";

newGuiPlayerView(ShowPlayer) {
               Profile = "GuiDefaultProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "-38 -15";
               Extent = "210 250";
MinExtent = "8 2";
               Visible = "1";
applyFilterToChildren = "1";
cameraZRot = "0";
forceFOV = "0";
            };
newGuiTextCtrl(AvatarText) {
               Profile = "GuiTextProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "8 0";
               Extent = "33 18";
MinExtent = "8 2";
               Visible = "1";
text = "Avatar";
maxLength = "255";
            };
         };
newGuiControl(RoleSectionBox) {
            Profile = "GuiWindowProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "160 70";
            Extent = "135 227";
MinExtent = "8 2";
            Visible = "1";
```

```
newGuiCheckBoxCtrl(CheckBoxPM) {
               Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 25";
               Extent = "103 30";
MinExtent = "8 2";
               Visible = "1";
               Command = "LoginDlg.clearRole(); a.setValue(1); $PlayerChoice
= \"green\"; LoginDlg.getModel();";
text = "Project Manager";
groupNum = "-1";
buttonType = "ToggleButton";
            };
newGuiCheckBoxCtrl(CheckBoxArch) {
               Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 55";
               Extent = "68 30";
MinExtent = "8 2";
               Visible = "1";
               Command = "LoginDlg.clearRole(); b.setValue(1); $PlayerChoice
= \"blue\"; LoginDlg.getModel();";
text = "Architect";
groupNum = "-1";
buttonType = "ToggleButton";
            };
newGuiTextCtrl(RoleText) {
               Profile = "GuiTextProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "8 0";
               Extent = "21 18";
MinExtent = "8 2";
               Visible = "1";
text = "Role";
maxLength = "255";
            };
newGuiCheckBoxCtrl(CheckBoxPM) {
               Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 85";
               Extent = "69 30";
MinExtent = "8 2";
               Visible = "1";
               Command = "LoginDlg.clearRole(); c.setValue(1); $PlayerChoice
= \"grey\"; LoginDlg.getModel();";
text = "Engineer";
groupNum = "-1";
buttonType = "ToggleButton";
            };
newGuiCheckBoxCtrl(CheckBoxCont) {
               Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
```

```
position = "15 115";
                Extent = "76 30";
MinExtent = "8 2";
                Visible = "1";
                Command = "LoginDlg.clearRole(); d.setValue(1); $PlayerChoice
= \"red\"; LoginDlg.getModel();";
text = "Contractor";
groupNum = "-1";
buttonType = "ToggleButton";
            };
newGuiCheckBoxCtrl(CheckBoxOwner) {
                Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 145";
                Extent = "58 30";
MinExtent = "8 2";
                Visible = "1";
                Command = "LoginDlg.clearRole(); e.setValue(1); $PlayerChoice
= \"yellow\"; LoginDlg.getModel();";
text = "Owner";
groupNum = "-1";
buttonType = "ToggleButton";
            };
newGuiCheckBoxCtrl(CheckBoxGuest) {
                Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 175";
                Extent = "55 30";
MinExtent = "8 2";
                Visible = "1";
                Command = "LoginDlg.clearRole(); f.setValue(1); $PlayerChoice
= \"white\"; LoginDlg.getModel();";
text = "Guest";
groupNum = "-1";
buttonType = "ToggleButton";
            };
         };
newGuiButtonCtrl(CreateSessionButton) {
            Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "474 314";
            Extent = "110 30";
MinExtent = "8 2";
            Visible = "1";
            Command = "canvas.setContent(startMissionGui);";
text = "Create Session";
groupNum = "-1";
buttonType = "PushButton";
         };
newGuiControl(UserInfoSectionBox) {
            Profile = "GuiWindowProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 70";
```

```
            Extent = "135 227";
MinExtent = "8 2";
            Visible = "1";

newGuiTextEditCtrl(PasswordBox) {
               Profile = "GuiTextEditProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "10 129";
               Extent = "115 18";
MinExtent = "8 2";
               Visible = "1";
maxLength = "16";
historySize = "0";
password = "1";
tabComplete = "0";
sinkAllKeyEvents = "0";
password = "1";
passwordMask = "*";
            };
newGuiTextCtrl(PleasEnterUserInfoText) {
               Profile = "GuiTextProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "9 0";
               Extent = "60 18";
MinExtent = "8 2";
               Visible = "1";
text = "Please Enter";
maxLength = "255";
            };
newGuiTextCtrl(PsswordText) {
               Profile = "GuiTextProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "10 101";
               Extent = "50 18";
MinExtent = "8 2";
               Visible = "1";
text = "Password";
maxLength = "255";
            };
newGuiTextCtrl(UsernameText) {
               Profile = "GuiTextProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "9 34";
               Extent = "49 18";
MinExtent = "8 2";
               Visible = "1";
text = "Username";
maxLength = "255";
            };
newGuiTextEditCtrl(UsernameBox) {
               Profile = "GuiTextEditProfile";
HorizSizing = "right";
VertSizing = "bottom";
```

```
position = "10 59";
               Extent = "115 18";
MinExtent = "8 2";
               Visible = "1";
text = "Guest";
maxLength = "16";
historySize = "0";
password = "0";
tabComplete = "0";
sinkAllKeyEvents = "0";
password = "0";
passwordMask = "*";
            };
         };
newGuiTextCtrl(VDRSTitleText) {
            Profile = "GuiMediumTextProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "160 32";
            Extent = "281 28";
MinExtent = "8 2";
            Visible = "1";
text = "Virtual Design Review System";
maxLength = "255";
         };
newGuiControl(SystemSelectionSectionBox) {
            Profile = "GuiWindowProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "305 70";
            Extent = "135 227";
MinExtent = "8 2";
            Visible = "1";

newGuiTextCtrl(SystemText) {
               Profile = "GuiTextProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "8 0";
               Extent = "36 18";
MinExtent = "8 2";
               Visible = "1";
text = "System";
maxLength = "255";
            };
newGuiCheckBoxCtrl(SysSelectArch) {
               Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 25";
               Extent = "108 30";
MinExtent = "8 2";
               Visible = "1";
               Command = "LoginDlg.clearSystem(); j.setValue(1);";
text = "Architectural (All)";
groupNum = "-1";
buttonType = "ToggleButton";
```

```
            };
newGuiCheckBoxCtrl(SysSelectStruc) {
               Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 55";
               Extent = "75 30";
MinExtent = "8 2";
               Visible = "1";
               Command = "LoginDlg.clearSystem(); k.setValue(1);";
text = "Structural";
groupNum = "-1";
buttonType = "ToggleButton";
            };
newGuiCheckBoxCtrl(SysSelectCivil) {
               Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 85";
               Extent = "75 30";
MinExtent = "8 2";
               Visible = "1";
               Command = "LoginDlg.clearSystem(); l.setValue(1);";
text = "Civil";
groupNum = "-1";
buttonType = "ToggleButton";
            };
newGuiCheckBoxCtrl(SysSelectMech) {
               Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 115";
               Extent = "81 30";
MinExtent = "8 2";
               Visible = "1";
               Command = "LoginDlg.clearSystem(); m.setValue(1);";
text = "Mechanical";
groupNum = "-1";
buttonType = "ToggleButton";
            };
newGuiCheckBoxCtrl(SysSelectElec) {
               Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 145";
               Extent = "81 30";
MinExtent = "8 2";
               Visible = "1";
               Command = "LoginDlg.clearSystem(); n.setValue(1);";
text = "Electrical";
groupNum = "-1";
buttonType = "ToggleButton";
            };
newGuiCheckBoxCtrl(SysSelectPlum) {
               Profile = "GuiCheckBoxProfile";
HorizSizing = "right";
VertSizing = "bottom";
```

```
position = "15 175";
                Extent = "83 30";
MinExtent = "8 2";
                Visible = "1";
                Command = "LoginDlg.clearSystem(); o.setValue(1);";
text = "Plumbing";
groupNum = "-1";
buttonType = "ToggleButton";
            };
         };
newGuiButtonCtrl(ExitButton) {
            Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "15 314";
            Extent = "110 30";
MinExtent = "8 2";
            Visible = "1";
            Command = "quit();";
text = "Exit";
groupNum = "-1";
buttonType = "PushButton";
         };
newGuiButtonCtrl(OptionsButton) {
            Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "246 314";
            Extent = "110 30";
MinExtent = "8 2";
            Visible = "1";
            Command = "Canvas.pushDialog(optionsDlg);";
text = "Options";
groupNum = "-1";
buttonType = "PushButton";
         };
newGuiButtonCtrl(RecordingsButton) {
            Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "131 314";
            Extent = "110 30";
MinExtent = "8 2";
            Visible = "1";
            Command = "Canvas.pushDialog(recordingsDlg);";
text = "Recordings";
groupNum = "-1";
buttonType = "PushButton";
         };
      };
   };
};
//--- OBJECT WRITE END ---
```

Appendix 8: Codes for the 3D Object Manipulation and Right-Click Menu System

worldEditor.h header file

```
bool        MenuUp;           // for right click object menu
bool        SpecsUp;          // left click specs
Point2F     lastMousePos;     // for click object menu
bool        ClientOnly;       // is it running in client only mode?
bool        mPlanarMovement;
S32         mUndoLimit;
S32         mDropType;
```

worldEditor.cc file

```
WorldEditor::WorldEditor()
{
// initialize the field data
MenuUp=false;         // initialize variable


void WorldEditor::on3DRightMouseDown(const Gui3DMouseEvent &event)
{
// START
   lastMousePos= Point2F(event.mousePoint.x, event.mousePoint.y);
if(MenuUp)
      Con::executef(Canvas, 2, "popDialog", "PopupMenu");
   MenuUp=false;
if(SpecsUp)                         // start of specifications menu
      Con::executef(Canvas, 2, "popDialog", "SpecsMenu");
   SpecsUp=false;                   // end of specifications menu code
   // END
}

void WorldEditor::on3DRightMouseUp(const Gui3DMouseEvent &event)
{
// START
if(mHitObject && !MenuUp)
      {MenuUp=true;
       mSelected.clear();
       mSelected.addObject(mHitObject);
         Con::executef(Canvas, 2, "pushDialog", "PopupMenu");
      }
   Con::executef(Canvas, 2, "popDialog", "alterObject");  // start alter
object menu
   Con::executef(Canvas, 2, "pushDialog", "alterObject"); // end alter object
menu code
   // END
}

void WorldEditor::on3DMouseDown(const Gui3DMouseEvent &event)
{
```

```
   mMouseDown = true;
   mMouseDragged = false;
   mLastRotation = 0.f;

mouseLock();

if(SpecsUp)           // remove SpecsMenu and PopupMenu if they are up
        Con::executef(Canvas, 2, "popDialog", "SpecsMenu");
if(MenuUp)
      Con::executef(Canvas, 2, "popDialog", "PopupMenu");
   SpecsUp=false;
   MenuUp=false;
```

<u>New code implemented for the PopupMenu.gui file</u>

```
//--- OBJECT WRITE BEGIN ---
newGuiControl(PopupMenu) {
   Profile = "GuiModelessDialogProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "0 0";
   Extent = "1024 768";
MinExtent = "8 2";
   Visible = "1";

newGuiWindowCtrl(PopupPane) {
      Profile = "GuiWindowProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "511 383";
      Extent = "103 248";
MinExtent = "8 2";
      Visible = "1";
maxLength = "255";
resizeWidth = "0";
resizeHeight = "0";
canMove = "0";
canClose = "0";
canMinimize = "0";
canMaximize = "0";
minSize = "50 50";

newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 185";
         Extent = "100 21";
MinExtent = "8 2";
         Visible = "1";
         Command = "if($IsClient)
DeleteObject(EWorldEditor.getSelectedObject(0)); else
EWorldEditor.deleteSelection(); Canvas.popDialog(PopupMenu);";
text = "Delete Object";
groupNum = "-1";
buttonType = "PushButton";
```

```
      };
newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 225";
         Extent = "100 21";
MinExtent = "8 2";
         Visible = "1";
         Command = "EWorldEditor.copySelection();
EWorldEditor.pasteSelection(); Canvas.popDialog(PopupMenu);";
text = "Duplicate Object";
groupNum = "-1";
buttonType = "PushButton";
      };
newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 205";
         Extent = "100 21";
MinExtent = "8 2";
         Visible = "1";
         Command = "EWorldEditor.copySelection();
Canvas.popDialog(PopupMenu);";
text = "Copy Object";
groupNum = "-1";
buttonType = "PushButton";
      };
newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 60";
         Extent = "100 16";
MinExtent = "8 2";
         Visible = "1";
         Command = "Canvas.popDialog(PopupMenu);";
text = "-----";
groupNum = "-1";
buttonType = "PushButton";
      };
newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 1";
         Extent = "100 21";
MinExtent = "8 2";
         Visible = "1";
         Command = "EWorldEditor.setMode(\"move\");
CurrentMode.setValue(\"Mode: Move\"); Canvas.popDialog(PopupMenu);
Canvas.popDialog(alterObject); Canvas.pushDialog(alterObject);";
text = "Move Mode";
groupNum = "-1";
buttonType = "PushButton";
      };
```

```
newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 21";
         Extent = "100 21";
MinExtent = "8 2";
         Visible = "1";
         Command = "EWorldEditor.setMode(\"rotate\");
CurrentMode.setValue(\"Mode: Rotate\"); Canvas.popDialog(PopupMenu);
Canvas.popDialog(alterObject); Canvas.pushDialog(alterObject);";
text = "Rotate Mode";
groupNum = "-1";
buttonType = "PushButton";
      };
newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 41";
         Extent = "100 21";
MinExtent = "8 2";
         Visible = "1";
         Command = "EWorldEditor.setMode(\"scale\");
CurrentMode.setValue(\"Mode: Scale\"); Canvas.popDialog(PopupMenu);
Canvas.popDialog(alterObject); Canvas.pushDialog(alterObject);";
text = "Scale Mode";
groupNum = "-1";
buttonType = "PushButton";
      };
newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 115";
         Extent = "100 16";
MinExtent = "8 2";
         Visible = "1";
         Command = "Canvas.popDialog(PopupMenu);";
text = "-----";
groupNum = "-1";
buttonType = "PushButton";
      };
newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 95";
         Extent = "100 21";
MinExtent = "8 2";
         Visible = "1";
         Command = "ResetScale(EWorldEditor.getSelectedObject(0));";
text = "Reset Scale";
groupNum = "-1";
buttonType = "PushButton";
      };
newGuiButtonCtrl() {
```

```
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 75";
         Extent = "100 21";
MinExtent = "8 2";
         Visible = "1";
         Command = "ResetRotation(EWorldEditor.getSelectedObject(0));";
text = "Reset Rotation";
groupNum = "-1";
buttonType = "PushButton";
      };
newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 130";
         Extent = "100 21";
MinExtent = "8 2";
         Visible = "1";
         Command = "(EWorldEditor.getSelectedObject(0)).hideObject();
Canvas.popDialog(PopupMenu);";
text = "Hide Object";
groupNum = "-1";
buttonType = "PushButton";
      };
newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 170";
         Extent = "100 16";
MinExtent = "8 2";
         Visible = "1";
         Command = "Canvas.popDialog(PopupMenu);";
text = "-----";
groupNum = "-1";
buttonType = "PushButton";
      };
newGuiButtonCtrl() {
         Profile = "GuiButtonProfile";
HorizSizing = "right";
VertSizing = "bottom";
position = "1 150";
         Extent = "100 21";
MinExtent = "8 2";
         Visible = "1";
         Command = "MissionGroup.showAll(); Canvas.popDialog(PopupMenu);";
text = "Show All Objects";
groupNum = "-1";
buttonType = "PushButton";
      };
   };
};
```

netConnection.h

```
bool isGhostingFrom()
    {bool ClientOnly  = dAtob(Con::getVariable("$IsClient"));  // Start
if(ClientOnly)                                       // hard coded avoid
crash
      return 1;                                                // End
return mGhostArray != NULL;
    };
```

netGhost.cc

```
#include"console/consoleTypes.h"
#include"editor/worldEditor.h"

#define DebugChecksum 0xF00DBAAD

extern U32 gGhostUpdates;
int updateSpeed = 5;


void NetConnection::ghostWritePacket(BitStream *bstream, PacketNotify
*notify)
{
#ifdef    TORQUE_DEBUG_NET
   bstream->writeInt(DebugChecksum, 32);
#endif

// START for adjustable client update speeds
if(!dAtob(Con::getVariable("$IsClient")))
      {
       updateSpeed--;
if(updateSpeed < 0)
          {updateSpeed = dAtoi(Con::getVariable("$clientUpdateSpeed"));
           Con::executef(1, "updateFromClient");
          }
      }
// END

...

AssertISV(mLocalGhosts[index] != NULL, "Invalid dest ghost.");
            AssertISV(origId == mLocalGhosts[index]-
>getClassId(getNetClassGroup()),
               avar("class id mismatch for dest class %s.",
                  mLocalGhosts[index]->getClassName()) );
#endif
// START ADD this fixes the client and the server 'fighting' over an object
      Con::executef(2, "isMoving", Con::getIntArg(mLocalGhosts[index]-
            >getId()));
            bool movin = dAtob(Con::getVariable("$IsMoving"));
            bool force = dAtob(Con::getVariable("$ForceUpdate"));
            if(mLocalGhosts[index]->getClassName() == "InteriorInstance")
                  add exceptions to rotate crash here
```

```
            force = 0;
            if(!movin || force) // don’t update the object if it is currently
            being moved on the client side=p
mLocalGhosts[index]->unpackUpdate(this, bstream); // will crash
                  if this doesn't execute when scaling or rotating
      // END ADD
         }

S32 NetConnection::getGhostIndex(NetObject *obj)
{  bool ClientOnly  = dAtob(Con::getVariable("$IsClient"));  // Start
//Client shouldn't be able to call this
if(ClientOnly)
return -1;                                             // End
```

sceneObject.h

```
// Functions for object hidding and client update
bool isShown();
void hideObject();
void unhideObject();
virtualvoid setTransformByClient(const MatrixF & mat);
virtualvoid setScaleByClient(const VectorF & scale);
virtualvoid updateFromClients();
// END

/// Disables collisions for this object including raycasts
      virtualvoid disableCollision();
```

```
LightingInfo      mLightingInfo; // Lighting info for this object
bool              transDirty;    //
bool              scaleDirty;    //
   MatrixF           newTrans;   //
   Point3F           newScale;   //
```

```
   Point3F mObjScale;  // Object scale

Point3F mOldScale;     // for object hide/unhide
bool    mShow;         //
```

sceneObject.cc

```
// START - JUST SOME SIMPLE EXPOSURES TO THE CONSOLE

ConsoleMethod( SceneObject, mShow, bool, 2, 2, "") //
{
return object->isShown();
}

ConsoleMethod( SceneObject, inspectPostApply, void, 2, 2, "") //
{
   object->inspectPostApply();
}

ConsoleMethod( SceneObject, hideObject, void, 2, 2, "") //
{
   object->hideObject();
}

ConsoleMethod( SceneObject, unhideobject, void, 2, 2, "") //
{
   object->unhideObject();
}

ConsoleMethod( SceneObject, setHidden, void, 3, 3, "(bool show)") //
{
   object->setHidden(dAtob(argv[2]));
}

ConsoleMethod( SceneObject, isHidden, bool, 2, 2, "") //
{
return object->isHidden();
}

ConsoleMethod( SceneObject, setTransformByClient, void, 3, 3, "(Transform
T)") //
{
   Point3F pos;
const MatrixF& tmat = object->getTransform();
   tmat.getColumn(3,&pos);
   AngAxisF aa(tmat);

   dSscanf(argv[2],"%g %g %g %g %g %g %g",
&pos.x,&pos.y,&pos.z,&aa.axis.x,&aa.axis.y,&aa.axis.z,&aa.angle);

   MatrixF mat;
   aa.setMatrix(&mat);
   mat.setColumn(3,pos);
   object->setTransformByClient(mat);
}

ConsoleMethod( SceneObject, setScaleByClient, void, 3, 3, "(Point3F scale)")
//
{
   VectorF scale(0.f,0.f,0.f);
   dSscanf(argv[2], "%g %g %g", &scale.x, &scale.y, &scale.z);
```

```
   object->setScaleByClient(scale);
}
// END
```

```
SceneObject::SceneObject()
{
      mShow = 1;                     // init vars
      transDirty = false;            // client update flag
      scaleDirty = false;            // client update flag
        mOldScale = Point3F(1, 1, 1);  // save scale before hide
```

```
constchar* SceneObject::scriptThis()
{
return Con::getIntArg(getId());
}



bool SceneObject::isShown() //
{return (mShow);
}

void SceneObject::hideObject() //
{if(mShow!=1)
    {//Con::printf("Object already invisible.");
      return;
    }
//Con::printf("Object %d hidden.", getId());
 mShow=0;

if(this->getClassName() == "AudioEmitter") //Get rid of those damn floating
things
      Con::executef(this, 1, "hideAudioData");
 mOldScale=mObjScale;
 mObjScale.set(0.0000000001, 0.0000000001, 0.0000000001); //Most object are
no longer rendered even at .001 scale
 setHidden(1);
 setLocked(1);
 inspectPostApply();
}

void SceneObject::unhideObject() //
{if(mShow!=0)
      {//Con::printf("Object already visible.");
      return;
    }
//Con::printf("Object %d unhidden.", getId());
 mShow=1;
 setLocked(0);
if(this->getClassName() == "AudioEmitter")
    Con::executef(this, 1, "unhideAudioData");
 mObjScale=mOldScale;
 setHidden(0);
```

```
 inspectPostApply();
}

// END
```

```
void SceneObject::removeFromScene()
{
if (mSceneManager != NULL)
      mSceneManager->removeObjectFromScene(this);
if (getContainer())
      getContainer()->removeObject(this);
}

// START ADD
void SceneObject::setTransformByClient(const MatrixF& mat)
{
this->transDirty = true;
this->newTrans = mat;
}

void SceneObject::setScaleByClient(const VectorF & scale)
{
this->scaleDirty = true;
this->newScale = scale;
}

void SceneObject::updateFromClients()
{
}
// END ADD
```

Appendix 9: Codes for Colaborative Design Review across Network

\client\scripts\clientEditing.cs

```
function serverSide(%soughtId)
{
for(%i=0; %i < ServerConnection.getCount(); %i++)
if((ServerConnection.getObject(%i)).getId()==%soughtId)
        {$WasInGroup=1;
return 1;
        }
 $WasInGroup=0;
return 0;
}

function PasteObjects()//
{
 commandToServer('PasteObjects', EWorldEditor.getScreenCenter());
}

function MoveObject(%id, %trans)  //set new position of moving object
{
 $moveList[%id.getId()]=1;
if(%trans$="%d")
    %trans=%id.getTransform();
 %id2=%id.getGhostId();
//echo("You are now moving " @ %id @ " which is " @ %id2 @ " as a GhostId");
 CommandToServer('MoveObject', %id2, %trans);
}

function ScaleObject(%id, %scale)//
{
 $moveList[%id.getId()]=1;
if(%scale$="%d")
    %scale=%id.getScale();
//echo("id: " @ %id @ " Ghost id: " @ %id.getGhostId() @ " Scale: " @
%scale);
 CommandToServer('ScaleObject', %id.getGhostId(), %scale);
}

function ResetRotation(%id)  //set rotation to none
{
if($IsClient)
    {EWorldEditor.resetRotation();
     %size = EWorldEditor.getSelectionSize();
     %obj = 0;
while(%obj < %size)
        {%LastObject = EWorldEditor.getSelectedObject(%obj);
         MoveObject(%LastObject, (%LastObject.getTransform()));
         %obj++;
        }
    }
else
```

```
    EWorldEditor.resetRotation();

 Canvas.popDialog(PopupMenu);
}

function ResetScale(%id)  // set scale to 1
{
 %size = EWorldEditor.getSelectionSize();
 %obj = 0;
while(%obj < %size)
    {
     %LastObject = (EWorldEditor.getSelectedObject(%obj));
if($IsClient)
        ScaleObject(%LastObject, "1 1 1");
else
        {%LastObject.setScale("1 1 1");
         %LastObject.inspectPostApply();
        }
     %obj++;
    }

 Canvas.popDialog(PopupMenu);
}

function ApplyTransforms(%id)  // set rotation
{
 %id2=%id.getGhostId();
 CommandToServer('ApplyTransforms', %id2);
}

function ClearMoveList()  // clear objects that are flagged as moving
{
 deleteVariables("$" @ moveList @ "*");
}

function clientSaveRequest()
{
 CommandToServer('ClientSave');
}

END FUNCTION ADDS
```

\server\scripts\commands.cs

```
// START FUNCTION ADDS

function serverCmdClearCopyList(%who)
{echo(%who @ " has cleared the copy list");
{
deleteVariables("$copy" @ %who @ "*");
$copysize[%who]=0;
}

function serverCmdCopyObject(%who, %id)
{echo(%who @ " has copied object " @ %who.resolveObjectFromGhostIndex(%id));
{
$copy[%who @ $copysize[%who]]=%who.resolveObjectFromGhostIndex(%id);
$copysize[%who]++;
}

function serverCmdPasteObjects(%who, %pos)
{echo(%who @ " has pasted his selection to pos ");
{
EWorldEditor.clientPaste(%who, getWord(%pos, 0), getWord(%pos, 1),
getWord(%pos, 2));
}

function getCopySize(%who)
{ $tempSize = $copysize[%who];
}

function getSingleObj(%who, %size)
{ $anObj = $copy[(%who @ %size)];
}

function serverCmdMoveObject(%who, %id, %newTransform)  // command to server
to move object from client side
{
if(%id $= "-1")
return;
 %serverObj = %who.resolveObjectFromGhostIndex(%id);
if(%serverObj $= "-1") return;
echo(%who @ " is moving " @ %id @ " which is " @ %serverObj @ " on server
side...");
 %serverObj.setTransformByClient(%newTransform);
 EWorldEditor.addToDirty(%serverObj);
 %serverObj.inspectPostApply();
}

function serverCmdScaleObject(%who, %id, %scale)  // command to server to
scale object
{
if(%id $= "-1")
return;
 %serverObj = %who.resolveObjectFromGhostIndex(%id);
if(%serverObj $= "-1")
return;
echo("server id: " @ %serverObj @ " Ghost id: " @ %id @ " Scale: " @ %scale);
```

```
 %serverObj.setScaleByClient(%scale);
 EWorldEditor.addToDirty(%serverObj);
 %serverObj.inspectPostApply();
}

function serverCmdDeleteObject(%who, %id)  // command to server to delete
object
{
 %serverObj = %who.resolveObjectFromGhostIndex(%id);
 %serverObj.delete();
 %serverObj.inspectPostApply();
}

function serverCmdClientHide(%who, %id)  // command to server to delete
object
{
 %serverObj = %who.resolveObjectFromGhostIndex(%id);
 %serverObj.hideObject();
 %serverObj.inspectPostApply();
}

function serverCmdClientUnhide(%who, %id)  // command to server to delete
object
{
 %serverObj = %who.resolveObjectFromGhostIndex(%id);
 %serverObj.unhideObject();
 %serverObj.inspectPostApply();
}

function serverCmdClientUnhideAll(%who, %id)  // command to server to delete
object
{
if(%id $= "MG")  //clients don't have MissionGroup
    %serverObj = MissionGroup.getId();
else
    %serverObj = %who.resolveObjectFromGhostIndex(%id);

 %serverObj.showAll();
}

function serverCmdApplyTransforms(%who, %id) // command to server to rotate
object
{
 %serverObj = %who.resolveObjectFromGhostIndex(%id);
 %serverObj.inspectPostApply();
}

function serverCmdChangePOV(%who, %cmd, %atPlayer)  // Server must change who
the camera is following
{
if(%cmd $= "ThirdPersonMode")
     %who.getCameraObject().setThirdPersonMode();
if(%cmd $= "OrbitMode")
     {%i=0;
//echo("%-----------atPlayer = ");
      echo(%atPlayer);
//first find where %who is
```

```
while(!(ClientGroup.getObject(%i).getControlObject() $=
%who.getControlObject()) )
            {%i++;
if(%i >= 129)//if in infinite loop break out. this should never happen.
should...
break;
            }
if(%cmd $= "GodViewMode")
      %who.getCameraObject().setGodViewMode();
}

function serverCmdChangeCamSpeed(%who, %speed)  // change camera speed
{
  $Camera::movementSpeed = %speed;
}

function serverCmdHideAllAudio(%who)  // Hide all audio emitter
{
  MissionGroup.hideAllAudio();
}

function serverCmdUnhideAllAudio(%who)  // Unhide all audio emitter
{
  MissionGroup.unhideAllAudio();
}

function updateFromClient()  //
{
  EWorldEditor.updateFromClient();
}

function SimSet::clientSave(%this, %who)  //Called on MissionGroup first
{
//echo("Create '" @ %this.getName() @ "' SimGroup on client Now.");
if(%this.getClassName() $= "SimGroup")
     commandtoclient(%who, 'CheckSimGroup', %this.getName(),
%this.getGroup().getName());
else
     commandtoclient(%who, 'CheckPath', %this.getName(),
%this.getGroup().getName(), %this.isLooping);

for(%i=0; %i < %this.getCount(); %i++)
if(!( ((%this.getObject(%i)).getClassName()) $= "SimGroup" || (
((%this.getObject(%i)).getClassName()) $= "Path")))  // not SimGroup
        {if((%this.getObject(%i)).getName() $= "MissionInfo")
            commandtoclient(%who, 'CreateMissionInfo', MissionInfo.name,
MissionInfo.desc0);
else
            commandtoclient(%who, 'SetObjectsGroup',
%who.getGhostId(%this.getObject(%i)), %this.getName());
//echo("Add '" @ %this.getObject(%i) @ "' to '" @ %this.getName() @ "' on
client Now.");
        }
else//if it is another SimGroup call recursively
        (%this.getObject(%i)).clientSave(%who);
}
// END FUNCTION ADDS
```

creator\editor\editor.cs

```
function toggleEditor(%make)
{
if (%make)
   {
if (!$missionRunning)
      {

if (!(Canvas.getContent() == EditorGui.getId()))
         {
            Editor::create();
            MissionCleanup.add(Editor);
            EditorGui.loadingMission = true;
            EditorGui.saveAs = true;
            Editor.open();
         }
else
            Editor.close("PlayGui");

         $dropcameracount = 0;
//schedule(100,0,dropFreakinCameraAtPlayer);

// END EDIT
```

Appendix 10: SQLite Integration Script

```
// This contains most of the functions for the SQLite Database handling


function DatabaseDlg::SaveText(%this, %dbname)
{
  %file = new FileObject();
  %file.OpenForWrite($workDir @ %dbname @ ".txt");

  %sqlite = new SQLiteObject(sqlite);

   // open database
   if (sqlite.openDatabase("LOG") == 0)                   // hardcoded
      {echo("ERROR: Failed to create database: " @ %dbname);
       echo("Ensure that the disk is not full or write protected. Function
aborted.");
       sqlite.delete();
       return;
      }

   %query = "SELECT * FROM head";
   %result = sqlite.query(%query, 0);
   if (%result == 0)
      echo("ERROR: No data found, table is empty.");

   %total="";
   // attempt to retrieve result data
   while (!sqlite.endOfResult(%result))
   {
      %tempData = sqlite.getColumn(%result, "data");
      %file.writeLine(%tempData);
      sqlite.nextRow(%result);
   }

   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();

   %file.delete();
}

function addEntry(%this)
{
 if(strcmp($EntryToAdd, ""))
    {appendDatabase($DatabaseName, $EntryToAdd);
     $entryToAdd="";
    }
 Canvas.popDialog(DatabaseAddEntry);
 Canvas.popDialog(Database);
 Canvas.pushDialog(Database);
}

function resultNumber(%dbname)
{
```

```
   %RowNumber=0;
   %sqlite = new SQLiteObject(sqlite);
   // open database
   if (sqlite.openDatabase(%dbname) == 0)
      {MessageBoxOK("Sorry", "Failed to open " @ $DatabaseName @ " Database.
Make sure\nthe disk is not full or write protected.");
       sqlite.delete();
       return;
      }

   %query = "SELECT * FROM head";
   %result = sqlite.query(%query, 0);
   if (%result == 0)
      MessageBoxOK("Sorry", "The " @ $DatabaseName @ " Database currently
contains no data.");

   // attempt to retrieve result data
   while (!sqlite.endOfResult(%result))
   {  %tempData=sqlite.getColumn(%result, "data");
      if($Search$="*")
          %RowNumber++;
      else
         {%test=0;
          %foo=1; // is it returning blank strings?
          %i=0;
          while(%test==0 && %foo==1)
             { %tempWord = getWord(%tempData, %i);
               if(%i>3)               // sometimes Torque will retrieve blank
words from formatting
                  if(%tempWord$="")
                     %foo=0;
               if($Search $= %tempWord)
                  %test=1;
               %i++;
             }
          if(%test==1)
              %RowNumber++;
         }
      sqlite.nextRow(%result);
   }

   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();

   return(%RowNumber);
}

function DatabaseAll::onwake(%this)
{
   if($Search $= "")
      $Search="*";

   //SearchingText.setText("Searching...");      //default text
   DBButton1.setText("Codes Database");
   DBButton2.setText("Lessons Learned Database");
   DBButton3.setText("Personnel Database");
```

```
   DBButton4.setText("Checklists Database");
   DBButton5.setText("Objects Database");
   DBButton6.setText("2D Drawings Database");
   DBButton7.setText("Standards Database");
   DBButton8.setText("Basis of Design Database");

   DBButton1.setText("Codes Database - " @ resultNumber("Codes") @ "
results");
   DBButton2.setText("Lessons Learned Database - " @
resultNumber("LessonsLearned") @ " results");
   DBButton3.setText("Personnel Database - " @ resultNumber("Personnel") @ "
results");
   DBButton4.setText("Checklists Database - " @ resultNumber("Checklists") @
" results");
   DBButton5.setText("Objects Database - " @ resultNumber("Objects") @ "
results");
   DBButton6.setText("2D Drawings Database - " @ resultNumber("2DDrawings") @
" results");
   DBButton7.setText("Standards Database - " @ resultNumber("Standards") @ "
results");
   DBButton8.setText("Basis of Design Database - " @
resultNumber("BasisOfDesign") @ " results");
   //SearchingText.setText("Done");

}

function Database::onwake(%this)
{
   if($Search $= "")
      $Search="*";
   %RowNumber=1;

   DatabaseHeader.setText($DatabaseName @ " Database");      // for most cases
the name of the database and title are the same
   if ($DatabaseName$="LessonsLearned")                      // start special
naming
      DatabaseHeader.setText("Lessons Learned Database");
   if ($DatabaseName$="2DDrawings")
      DatabaseHeader.setText("2D Drawings Database");
   if ($DatabaseName$="BasisOfDesign")
      DatabaseHeader.setText("Basis of Design Database");   // end special
naming
   DatabaseList.clearSelection();
   DatabaseList.clear();

   %sqlite = new SQLiteObject(sqlite);
   // open database
   if (sqlite.openDatabase($DatabaseName) == 0)            //hardcoded
      {MessageBoxOK("Sorry", "Failed to open " @ $DatabaseName @ " Database.
Make sure\nthe disk is not full or write protected.");
       sqlite.delete();
       return;
      }

   %query = "SELECT * FROM head";
   %result = sqlite.query(%query, 0);
   if (%result == 0)
```

```
      MessageBoxOK("Sorry", "The " @ $DatabaseName @ " Database currently
contains no data.");

   // attempt to retrieve result data
   while (!sqlite.endOfResult(%result))
   {  %tempData=sqlite.getColumn(%result, "data");
      if($Search$="*")
         {DatabaseList.addrow(%RowNumber, %tempData);
          %RowNumber++;
         }
      else
         {%test=0;
          %foo=1; //is it returning blank strings?
          %i=0;
          while(%test==0 && %foo==1)
             { %tempWord = getWord(%tempData, %i);
               if(%i>3)               // sometimes Torque will retrieve
blanks from formatting
                  if(%tempWord$="")
                     %foo=0;
               if($Search $= %tempWord)
                  %test=1;
               %i++;
             }
          if(%test==1)
             {DatabaseList.addrow(%RowNumber, %tempData);
              %RowNumber++;
             }
         }
      sqlite.nextRow(%result);
   }

   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();
}

function DatabaseDlg::onwake(%this)
{
  if($workDir$="")

{if(IsFile("../Example/projectOne/data/shapes/furniture/unavailable.jpg")) //
hacktastic  O_o
         $workDir="../Example/";
      if(IsFile("../Virt*/projectOne/data/shapes/furniture/unavailable.jpg"))
         $workDir="../Virt*/";
     }

   %sqlite = new SQLiteObject(sqlite);

   // open database
   if (sqlite.openDatabase("LOG") == 0)                   // hardcoded
      {echo("ERROR: Failed to create database: " @ %dbname);
       echo("Ensure that the disk is not full or write protected. Function
aborted.");
       sqlite.delete();
       return;
```

```
      }

   %query = "SELECT * FROM head";
   %result = sqlite.query(%query, 0);
   if (%result == 0)
      echo("ERROR: No data found, table is empty.");

   %total="";
   // attempt to retrieve result data
   while (!sqlite.endOfResult(%result))
   {
      %tempData = sqlite.getColumn(%result, "data");
      %total= %total @ "\n" @ (%tempData);
      sqlite.nextRow(%result);
   }
   DatabaseText.setText(%total);

   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();
}

function showDatabase(%dbname)
{
   %sqlite = new SQLiteObject(sqlite);

   // open database
   if (sqlite.openDatabase(%dbname) == 0)
      {echo("ERROR: Failed to create database: " @ %dbname);
       echo("Ensure that the disk is not full or write protected. Function
aborted.");
       sqlite.delete();
       return;
      }

   %query = "SELECT * FROM head";
   %result = sqlite.query(%query, 0);
   if (%result == 0)
      echo("ERROR: No data found, table is empty.");

   echo("");
   echo("------------------Begin "@%dbname@"-------------------------------
");
   // attempt to retrieve result data
   while (!sqlite.endOfResult(%result))
   {
      %tempData = sqlite.getColumn(%result, "data");
      echo(%tempData);
      sqlite.nextRow(%result);
   }
   echo("");
   echo("-------------------End "@%dbname@"--------------------------------
");
   echo("");
   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();
```

```
}


function clearDatabase(%dbname)
{
   %sqlite = new SQLiteObject(sqlite);

   // open database
   if (sqlite.openDatabase(%dbname) == 0)
      {echo("ERROR: Failed to create database: " @ %dbname);
       echo("Ensure that the disk is not full or write protected. Function
aborted.");
       sqlite.delete();
       return;
      }

   %result = sqlite.query("DROP TABLE head", 0);
   if (%result == 0)
      echo("ERROR: Failed to DROP table");

   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();
}

function getFieldOfDatabase(%dbname, %table, %i)
{  %returnData = NULL;
   if(%table $= "")
      return(0);
   %sqlite = new SQLiteObject(sqlite);

   // open database
   if (sqlite.openDatabase(%dbname) == 0)
      {echo("ERROR: Failed to create database: " @ %dbname);
       echo("Ensure that the disk is not full or write protected. Function
aborted.");
       sqlite.delete();
       return;
      }

   %query = "SELECT * FROM " @ %table;
   %result = sqlite.query(%query, 0);
   if (%result == 0)
      {sqlite.delete();
       return(NULL);
      }
      //echo("ERROR: No data found, table is empty.");

   // attempt to retrieve result data
   %j=0;
   while (!sqlite.endOfResult(%result))
   {  %j++;
      %tempData = sqlite.getColumn(%result, "data");
      sqlite.nextRow(%result);
      if(%j==%i)
         %returnData=%tempData;
   }
```

```
   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();
   return(%returnData);
}

function addFieldToDatabase(%dbname, %table)
{
   %sqlite = new SQLiteObject(sqlite);

   if (sqlite.openDatabase(%dbname) == 0)
      {echo("ERROR: Failed to open database: " @ %dbname);
       echo("Ensure that the disk is not full or write protected. Function
aborted.");
       sqlite.delete();
       return;
      }

   %query = "CREATE TABLE " @ %table @ " (data TEXT)";
   %result = sqlite.query(%query, 0);
   if (%result == 0)
      {echo("ERROR: Failed to create new table.  Function aborted.");
       sqlite.closeDatabase();
       sqlite.delete();
       return;
      }
   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();
}

function createDatabase(%dbname)
{
   %sqlite = new SQLiteObject(sqlite);

   if (sqlite.openDatabase(%dbname) == 0)
      {echo("ERROR: Failed to create database: " @ %dbname);
       echo("Ensure that the disk is not full or write protected. Function
aborted.");
       sqlite.delete();
       return;
      }

   %query = "CREATE TABLE head (data TEXT)";
   %result = sqlite.query(%query, 0);
   if (%result == 0)
      {echo("ERROR: Failed to create new table.  Function aborted.");
       sqlite.closeDatabase();
       sqlite.delete();
       return;
      }
   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();
}

function appendDatabase(%dbname, %dataToInsert)
```

```
{
   %sqlite = new SQLiteObject(sqlite);

   // open database
   if (sqlite.openDatabase(%dbname) == 0)
      {echo("ERROR: Failed to create database: " @ %dbname);
       echo("Ensure that the disk is not full or write protected.  Function
aborted.");
       sqlite.delete();
       return;
      }

   %query = "INSERT INTO head VALUES ('" @ %dataToInsert @ "')";
   %result = sqlite.query(%query, 0);
   if (%result == 0)
      {// query failed
       echo("ERROR: Failed to INSERT test data into table. Make sure table
has been created.");
       sqlite.closeDatabase();
       sqlite.delete();
       return;
      }

   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();
}

function deleteEntryOfDatabase(%dbname, %entryId)
{
   %sqlite = new SQLiteObject(sqlite);

   // open database
   if (sqlite.openDatabase(%dbname) == 0)
      {echo("ERROR: Failed to create database: " @ %dbname);
       echo("       Ensure that the disk is not full or write protected.
Function aborted.");
       sqlite.delete();
       return;
      }

   %entryText = DatabaseList.getRowTextById(%entryId);
   //DatabaseList.removeRowById(%entryId);              // done on refresh

   %query = "DELETE FROM head WHERE data = '" @ %entryText @ "'";
   %result = sqlite.query(%query, 0);

   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();

   Canvas.popDialog(Database);
   Canvas.pushDialog(Database);
}

function clearFieldOfDatabase(%dbname, %table)
{
```

```
   %sqlite = new SQLiteObject(sqlite);

   // open database
   if (sqlite.openDatabase(%dbname) == 0)
      {echo("ERROR: Failed to create database: " @ %dbname);
       echo("Ensure that the disk is not full or write protected. Function
aborted.");
       sqlite.delete();
       return;
      }

   %result = sqlite.query("DROP TABLE " @ %table, 0);
   if (%result == 0)
      echo("ERROR: Failed to DROP table");

   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();
}

function appendFieldOfDatabase(%dbname, %dataToInsert, %table)
{
   %sqlite = new SQLiteObject(sqlite);

   // open database
   if (sqlite.openDatabase(%dbname) == 0)
      {echo("ERROR: Failed to create database: " @ %dbname);
       echo("Ensure that the disk is not full or write protected. Function
aborted.");
       sqlite.delete();
       return;
      }

   %query = "INSERT INTO " @ %table @ " VALUES ('" @ %dataToInsert @ "')";
   %result = sqlite.query(%query, 0);
   if (%result == 0)
      {// query failed
       echo("ERROR: Failed to INSERT test data into table. Make sure table
has been created.");
       sqlite.closeDatabase();
       sqlite.delete();
       return;
      }

   sqlite.clearResult(%result);
   sqlite.closeDatabase();
   sqlite.delete();
}


// Below is the sample script i based most of the database handling off of

function sqliteTest(%dbname)
{

   %sqlite = new SQLiteObject(sqlite);
   if (%sqlite == 0)
```

```
   {
      echo("ERROR: Failed to create SQLiteObject. sqliteTest aborted.");
      return;
   }

   // open database
   if (sqlite.openDatabase(%dbname) == 0)
   {
      echo("ERROR: Failed to open database: " @ %dbname);
      echo("Ensure that the disk is not full or write protected.  sqliteTest
aborted.");
      sqlite.delete();
      return;
   }

   // create a new simple table for demonstration purposes
   %query = "CREATE TABLE users (username VARCHAR(20))";
   %result = sqlite.query(%query, 0);
   if (%result == 0)
   {
      // query failed
      echo("ERROR: Failed to create new table.  sqliteTest aborted.");
      sqlite.closeDatabase();
      sqlite.delete();
      return;
   }
   // result sets are persistant, and you can have more than one at any time.
This
   // allows for cross referencing data. However, each result set takes up
   // memory and large queries will take up a significant amount of memory.
   // If you no longer need a result set, you should clear it.
   sqlite.clearResult(%result);
   // insert some data into the table
   %query[0] = "INSERT INTO users VALUES ('john')";
   %query[1] = "INSERT INTO users VALUES ('gax')";
   %query[2] = "INSERT INTO users VALUES ('mark')";
   %query[3] = "INSERT INTO users VALUES ('ford')";
   for(%i=0; %i<=3; %i++)
     {%result = sqlite.query(%query[%i], 0);
      if (%result == 0)
         {
          // query failed
          echo("ERROR: Failed to INSERT test data into table.  Attempting to
DROP table before aborting.");
          if (sqlite.query("DROP TABLE users", 0) == 0)
             echo("Failed to DROP table.  sqliteTest aborted.");
          else
             echo("Table DROP completed properly.  sqliteTest aborted.");
          sqlite.closeDatabase();
          sqlite.delete();
          return;
         }
       sqlite.clearResult(%result);
      }
   // retrieve some data from the table
   %query = "SELECT * FROM users";
   %result = sqlite.query(%query, 0);
```

```
   if (%result == 0)
   {
      echo("ERROR: Failed to SELECT from users table.");
   }
   // attempt to retrieve result data
   while (!sqlite.endOfResult(%result))
   {
      %username = sqlite.getColumn(%result, "username");
      echo("Retrived username = " @ %username);
      sqlite.nextRow(%result);
   }

   sqlite.clearResult(%result);
   // delete the table
   %result = sqlite.query("DROP TABLE users", 0);
   if (%result == 0)
   {
      echo("ERROR: Failed to DROP table");
   }
   sqlite.clearResult(%result);
   // close database
   sqlite.closeDatabase();

   // delete SQLite object.
   sqlite.delete();
}

function sqlite::onQueryFailed(%this, %error)
{
   echo ("SQLite Query Error: " @ %error);
}
```

<u>The original C++ source code provided by John Vanderback is shown below</u>

```
//-----------------------------------------------------------------------------
// Torque Game Engine
// Written by John Vanderbeck
//
// This code is written by John Vanderbeck and is offered freely to the Torque
// Game Engine with no express warranties. Use it for whatever you want, all
// I ask is that you don't rip it off and call it your own. Credit where
// credit is due. If you do use this, just drop me a line to let me know.  It
// makes me feel good :)
// Contact: jvanderbeck@novusdelta.com
//          http://www.novusdelta.com
//
//-----------------------------------------------------------------------------
// This code implements support for SQLite into Torque and TorqueScript.
//
// Essentially this creates a scriptable object that interfaces with SQLite.
//-----------------------------------------------------------------------------

#include "console/SQLiteObject.h"

#include "console/simBase.h"
#include "console/consoleInternal.h"
#include <STDLIB.H>

IMPLEMENT_CONOBJECT(SQLiteObject);

SQLiteObject::SQLiteObject()
{
   m_pDatabase = NULL;
   m_szErrorString = NULL;
   m_iLastResultSet = 0;
   m_iNextResultSet = 1;
}

SQLiteObject::~SQLiteObject()
{
   S32 index;
   CloseDatabase();
   ClearErrorString();
   Vector<int> vTemp;
   Vector<int>::iterator iTemp;

   VectorPtr<sqlite_resultset*>::iterator i;
   for (i = m_vResultSets.begin(); i != m_vResultSets.end(); i++)
   {
      vTemp.push_back((*i)->iResultSet);
   }
   index = 0;
   for (iTemp = vTemp.begin(); iTemp != vTemp.end(); iTemp++)
   {
      Con::warnf("SQLiteObject Warning: Result set #%i was not cleared by
script.  Clearing it now.", vTemp[index]);
      ClearResultSet(vTemp[index]);
      index++;
   }
```

```
   m_vResultSets.clear();

}

bool SQLiteObject::processArguments(S32 argc, const char **argv)
{
   if(argc == 0)
      return true;
   else
      return true;

   return false;
}

bool SQLiteObject::onAdd()
{
   if (!Parent::onAdd())
      return false;

   const char *name = getName();
   if(name && name[0] && getClassRep())
   {
      Namespace *parent = getClassRep()->getNameSpace();
      Con::linkNamespaces(parent->mName, name);
      mNameSpace = Con::lookupNamespace(name);

   }

   return true;
}

// This is the function that gets called when an instance
// of your object is being removed from the system and being
// destroyed.  Use this to do your clean up and what not.
void SQLiteObject::onRemove()
{
   CloseDatabase();
   Parent::onRemove();
}

void SQLiteObject::initPersistFields()
{
   Parent::initPersistFields();
}

int Callback(void *pArg, int argc, char **argv, char **columnNames)
{
   // basically this callback is called for each row in the SQL query result.
   // for each row, argc indicates how many columns are returned.
   // columnNames[i] is the name of the column
   // argv[i] is the value of the column

   sqlite_resultrow* pRow;
   sqlite_resultset* pResultSet;
   char* name;
   char* value;
   int i;
```

```
    if (argc == 0)
        return 0;

    pResultSet = (sqlite_resultset*)pArg;
    if (!pResultSet)
        return -1;

    // create a new result row
    pRow = new sqlite_resultrow;
    pResultSet->iNumCols = argc;
    // loop through all the columns and stuff them into our row
    for (i = 0; i < argc; i++)
    {
        // DEBUG CODE
//      Con::printf("%s = %s\n", columnNames[i], argv[i] ? argv[i] : "NULL");
        name = new char[dStrlen(columnNames[i]) + 1];
        dStrcpy(name, columnNames[i]);
        pRow->vColumnNames.push_back(name);
        if (argv[i])
        {
            value = new char[dStrlen(argv[i]) + 1];
            dStrcpy(value, argv[i]);
            pRow->vColumnValues.push_back(value);
        }
        else
        {
            value = new char[10];
            dStrcpy(value, "NULL");
            pRow->vColumnValues.push_back(value);
        }
    }
    pResultSet->iNumRows++;
    pResultSet->vRows.push_back(pRow);

    // return 0 or else the sqlexec will be aborted.
    return 0;
}

bool SQLiteObject::OpenDatabase(const char* filename)
{
    // check to see if we already have an open database, and
    // if so, close it.
    CloseDatabase();

    // We persist the error string so that the script may make a
    // GetLastError() call at any time.  However when we get
    // ready to make a call which could result in a new error,
    // we need to clear what we have to avoid a memory leak.
    ClearErrorString();

    m_pDatabase = sqlite_open(filename, 0, &m_szErrorString);
    if (m_pDatabase == 0)
    {
        // there was an error and the database could not
        // be opened.
        Con::executef(this, 2, "onOpenFailed()", m_szErrorString);
```

```
      return false;
   }
   else
   {
      // database was opened without error
      Con::executef(this, 1, "onOpened()");
   }
   return true;
}

int SQLiteObject::ExecuteSQL(const char* sql)
{
   int iResult;
   sqlite_resultset* pResultSet;

   // create a new resultset
   pResultSet = new sqlite_resultset;

   if (pResultSet)
   {
      pResultSet->bValid = false;
      pResultSet->iCurrentColumn = 0;
      pResultSet->iCurrentRow = 0;
      pResultSet->iNumCols = 0;
      pResultSet->iNumRows = 0;
      pResultSet->iResultSet = m_iNextResultSet;
      pResultSet->vRows.clear();
      m_iLastResultSet = m_iNextResultSet;
      m_iNextResultSet++;
   }
   else
      return 0;

   iResult = sqlite_exec(m_pDatabase, sql, Callback, (void*)pResultSet,
&m_szErrorString);
   if (iResult == 0)
   {
      //SQLITE_OK
      SaveResultSet(pResultSet);
      Con::executef(this, 1, "onQueryFinished()");
      return pResultSet->iResultSet;
   }
   else
   {
      // error occured
      Con::executef(this, 2, "onQueryFailed", m_szErrorString);
      delete pResultSet;
      return 0;
   }

   return 0;
}

void SQLiteObject::CloseDatabase()
{
   if (m_pDatabase)
      sqlite_close(m_pDatabase);
```

```
   m_pDatabase = NULL;
}

void SQLiteObject::NextRow(int resultSet)
{
   sqlite_resultset* pResultSet;

   pResultSet = GetResultSet(resultSet);
   if (!pResultSet)
      return;

   pResultSet->iCurrentRow++;
}

bool SQLiteObject::EndOfResult(int resultSet)
{
   sqlite_resultset* pResultSet;

   pResultSet = GetResultSet(resultSet);
   if (!pResultSet)
      return true;

   if (pResultSet->iCurrentRow >= pResultSet->iNumRows)
      return true;

   return false;
}

void SQLiteObject::ClearErrorString()
{
   if (m_szErrorString)
      sqlite_freemem(m_szErrorString);

   m_szErrorString = NULL;
}

void SQLiteObject::ClearResultSet(int index)
{
   sqlite_resultset* resultSet;
   sqlite_resultrow* resultRow;
   S32 rows, cols, iResultSet;

   // Get the result set specified by index
   resultSet = GetResultSet(index);
   iResultSet = GetResultSetIndex(index);
   if ((!resultSet) || (!resultSet->bValid))
   {
      Con::warnf("Warning SQLiteObject::ClearResultSet(%i) failed to retrieve
specified result set.  Result set was NOT cleared.", index);
      return;
   }
   // Now we have the specific result set to be cleared.
   // What we need to do now is iterate through each "Column" in each "Row"
   // and free the strings, then delete the entries.
   VectorPtr<sqlite_resultrow*>::iterator iRow;
   VectorPtr<char*>::iterator iColumnName;
```

```
   VectorPtr<char*>::iterator iColumnValue;

   for (iRow = resultSet->vRows.begin(); iRow != resultSet->vRows.end();
iRow++)
   {
      // Iterate through rows
      // for each row iterate through all the column values and names
      for (iColumnName = (*iRow)->vColumnNames.begin(); iColumnName !=
(*iRow)->vColumnNames.end(); iColumnName++)
      {
         // Iterate through column names.  Free the memory.
         delete[] (*iColumnName);
      }
      for (iColumnValue = (*iRow)->vColumnValues.begin(); iColumnValue !=
(*iRow)->vColumnValues.end(); iColumnValue++)
      {
         // Iterate through column values.  Free the memory.
         delete[] (*iColumnValue);
      }
      // free memory used by the row
      delete (*iRow);
   }
   // empty the resultset
   resultSet->vRows.clear();
   resultSet->bValid = false;
   delete resultSet;
   m_vResultSets.erase_fast(iResultSet);
}

sqlite_resultset* SQLiteObject::GetResultSet(int iResultSet)
{
   // Get the result set specified by iResultSet
   VectorPtr<sqlite_resultset*>::iterator i;
   for (i = m_vResultSets.begin(); i != m_vResultSets.end(); i++)
   {
      if ((*i)->iResultSet == iResultSet)
         break;
   }

   return *i;
}

int SQLiteObject::GetResultSetIndex(int iResultSet)
{
   int iIndex;
   // Get the result set specified by iResultSet
   VectorPtr<sqlite_resultset*>::iterator i;
   iIndex = 0;
   for (i = m_vResultSets.begin(); i != m_vResultSets.end(); i++)
   {
      if ((*i)->iResultSet == iResultSet)
         break;
      iIndex++;
   }

   return iIndex;
}
```

```
bool SQLiteObject::SaveResultSet(sqlite_resultset* pResultSet)
{
   // Basically just add this to our vector.  It should already be filled up.
   pResultSet->bValid = true;
   m_vResultSets.push_back(pResultSet);

   return true;
}

int SQLiteObject::GetColumnIndex(int iResult, const char* columnName)
{
   int iIndex;
   VectorPtr<char*>::iterator i;
   sqlite_resultset* pResultSet;
   sqlite_resultrow* pRow;

   pResultSet = GetResultSet(iResult);
   if (!pResultSet)
      return 0;

   pRow = pResultSet->vRows[0];
   if (!pRow)
      return 0;

   iIndex = 0;
   for (i = pRow->vColumnNames.begin(); i != pRow->vColumnNames.end(); i++)
   {
      if (dStricmp((*i), columnName) == 0)
         return iIndex + 1;
      iIndex++;
   }

   return 0;
}

//-----------------------------------------------------------------------
// These functions are the code that actually tie our object into the
scripting
// language.  As you can see each one of these is called by script and in
turn
// calls the C++ class function.
ConsoleMethod(SQLiteObject, openDatabase, bool, 3, 3, "(const char* filename)
Opens the database specified by filename.  Returns true or false.")
{
   return object->OpenDatabase(argv[2]);
}

ConsoleMethod(SQLiteObject, closeDatabase, void, 2, 2, "Closes the active
database.")
{
   object->CloseDatabase();
}

ConsoleMethod(SQLiteObject, query, S32, 4, 0, "(const char* sql, int mode)
Performs an SQL query on the open database and returns an identifier to a
valid result set. mode is currently unused, and is reserved for future use.")
```

```
{
   S32 iCount;
   S32 iIndex, iLen, iNewIndex, iArg, iArgLen, i;
   char* szNew;

   if (argc == 4)
      return object->ExecuteSQL(argv[2]);
   else if (argc > 4)
   {
      // Support for printf type querys, as per Ben Garney's suggestion
      // Basically what this does is allow the user to insert questino marks
into thier query that will
      // be replaced with actual data.  For example:
      // "SELECT * FROM data WHERE id=? AND age<7 AND name LIKE ?"

      // scan the query and count the question marks
      iCount = 0;
      iLen = dStrlen(argv[2]);
      for (iIndex = 0; iIndex < iLen; iIndex++)
      {
         if (argv[2][iIndex] == '?')
            iCount++;
      }

      // now that we know how many replacements we have, we need to make sure
we
      // have enough arguments to replace them all.  All arguments above 4
should be our data
      if (argc - 4 == iCount)
      {
         // ok we have the correct number of arguments
         // so now we need to calc the length of the new query string.  This
is easily achieved.
         // We simply take our base string length, subtract the question
marks, then add in
         // the number of total characters used by our arguments.
         iLen = dStrlen(argv[2]) - iCount;
         for (iIndex = 1; iIndex <= iCount; iIndex++)
         {
            iLen = iLen + dStrlen(argv[iIndex+3]);
         }
         // iLen should now be the length of our new string
         szNew = new char[iLen];

         // now we need to replace all the question marks with the actual
arguments
         iLen = dStrlen(argv[2]);
         iNewIndex = 0;
         iArg = 1;
         for (iIndex = 0; iIndex <= iLen; iIndex++)
         {
            if (argv[2][iIndex] == '?')
            {
               // ok we need to replace this question mark with the actual
argument
               // and iterate our pointers and everything as needed.  This is
no doubt
```

```
                // not the best way to do this, but it works for me for now.
                // My god this is really a mess.
                iArgLen = dStrlen(argv[iArg + 3]);
                // copy first character
                szNew[iNewIndex] = argv[iArg + 3][0];
                // copy rest of characters, and increment iNewIndex
                for (i = 1; i < iArgLen; i++)
                {
                   iNewIndex++;
                   szNew[iNewIndex] = argv[iArg + 3][i];
                }
                iArg++;

             }
             else
                szNew[iNewIndex] = argv[2][iIndex];

             iNewIndex++;
          }
       }
       else
          return 0; // incorrect number of question marks vs arguments
       Con::printf("Old SQL: %s\nNew SQL: %s", argv[2], szNew);
       return object->ExecuteSQL(szNew);
    }

    return 0;
}

ConsoleMethod(SQLiteObject, clearResult, void, 3, 3, "(int resultSet) Clears
memory used by the specified result set, and deletes the result set.")
{
   object->ClearResultSet(dAtoi(argv[2]));
}

ConsoleMethod(SQLiteObject, nextRow, void, 3, 3, "(int resultSet) Moves the
result set's row pointer to the next row.")
{
   sqlite_resultset* pResultSet;
   pResultSet = object->GetResultSet(dAtoi(argv[2]));
   if (pResultSet)
   {
      pResultSet->iCurrentRow++;
   }
}

ConsoleMethod(SQLiteObject, previousRow, void, 3, 3, "(int resultSet) Moves
the result set's row pointer to the previous row")
{
   sqlite_resultset* pResultSet;
   pResultSet = object->GetResultSet(dAtoi(argv[2]));
   if (pResultSet)
   {
      pResultSet->iCurrentRow--;
   }
}
```

```
ConsoleMethod(SQLiteObject, firstRow, void, 3, 3, "(int resultSet) Moves the
result set's row pointer to the very first row in the result set.")
{
   sqlite_resultset* pResultSet;
   pResultSet = object->GetResultSet(dAtoi(argv[2]));
   if (pResultSet)
   {
      pResultSet->iCurrentRow = 0;
   }
}

ConsoleMethod(SQLiteObject, lastRow, void, 3, 3, "(int resultSet) Moves the
result set's row pointer to the very last row in the result set.")
{
   sqlite_resultset* pResultSet;
   pResultSet = object->GetResultSet(dAtoi(argv[2]));
   if (pResultSet)
   {
      pResultSet->iCurrentRow = pResultSet->iNumRows - 1;
   }
}

ConsoleMethod(SQLiteObject, setRow, void, 4, 4, "(int resultSet int row)
Moves the result set's row pointer to the row specified.  Row indices start
at 1 not 0.")
{
   sqlite_resultset* pResultSet;
   pResultSet = object->GetResultSet(dAtoi(argv[2]));
   if (pResultSet)
   {
      pResultSet->iCurrentRow = dAtoi(argv[3]) - 1;
   }
}

ConsoleMethod(SQLiteObject, getRow, S32, 3, 3, "(int resultSet) Returns what
row the result set's row pointer is currently on.")
{
   sqlite_resultset* pResultSet;
   pResultSet = object->GetResultSet(dAtoi(argv[2]));
   if (pResultSet)
   {
      return pResultSet->iCurrentRow + 1;
   }
   else
      return 0;
}

ConsoleMethod(SQLiteObject, numRows, S32, 3, 3, "(int resultSet) Returns the
number of rows in the result set.")
{
   sqlite_resultset* pResultSet;
   pResultSet = object->GetResultSet(dAtoi(argv[2]));
   if (pResultSet)
   {
      return pResultSet->iNumRows;
   }
   else
```

```
      return 0;
}

ConsoleMethod(SQLiteObject, numColumns, S32, 3, 3, "(int resultSet) Returns
the number of columns in the result set.")
{
   sqlite_resultset* pResultSet;
   pResultSet = object->GetResultSet(dAtoi(argv[2]));
   if (pResultSet)
   {
      return pResultSet->iNumCols;
   }
   else
      return 0;
}

ConsoleMethod(SQLiteObject, endOfResult, bool, 3, 3, "(int resultSet) Checks
to see if the internal pointer for the specified result set is at the end,
indicating there are no more rows left to read.")
{
   return object->EndOfResult(dAtoi(argv[2]));
}

ConsoleMethod(SQLiteObject, EOR, bool, 3, 3, "(int resultSet) Same as
endOfResult().")
{
   return object->EndOfResult(dAtoi(argv[2]));
}

ConsoleMethod(SQLiteObject, EOF, bool, 3, 3, "(int resultSet) Same as
endOfResult().")
{
   return object->EndOfResult(dAtoi(argv[2]));
}

ConsoleMethod(SQLiteObject, getColumnIndex, S32, 4, 4, "(resultSet
columnName) Looks up the specified column name in the specified result set,
and returns the columns index number.  A return value of 0 indicates the
lookup failed for some reason (usually this indicates you specified a column
name that doesn't exist or is spelled wrong).")
{
   return object->GetColumnIndex(dAtoi(argv[2]), argv[3]);
}

ConsoleMethod(SQLiteObject, getColumnName, const char *, 4, 4, "(resultSet
columnIndex) Looks up the specified column index in the specified result set,
and returns the column's name.  A return value of an empty string indicates
the lookup failed for some reason (usually this indicates you specified a
column index that is invalid or exceeds the number of columns in the result
set). Columns are index starting with 1 not 0")
{
   sqlite_resultset* pResultSet;
   sqlite_resultrow* pRow;
   VectorPtr<char*>::iterator iName;
   VectorPtr<char*>::iterator iValue;
   S32 iColumn;
```

```
   pResultSet = object->GetResultSet(dAtoi(argv[2]));
   if (pResultSet)
   {
      pRow = pResultSet->vRows[pResultSet->iCurrentRow];
      if (!pRow)
         return "";

      // We assume they specified column by index.  If they know the column
name they wouldn't be calling this function :)
      iColumn = dAtoi(argv[3]);
      if (iColumn == 0)
         return "";  // column indices start at 1, not 0

      // now we should have an index for our column name
      if (pRow->vColumnNames[iColumn])
         return pRow->vColumnNames[iColumn];
      else
         return "";
   }
   else
      return "";
}

ConsoleMethod(SQLiteObject, getColumn, const char *, 4, 4, "(resultSet
column) Returns the value of the specified column (Column can be specified by
name or index) in the current row of the specified result set. If the call
fails, the returned string will indicate the error.")
{
   sqlite_resultset* pResultSet;
   sqlite_resultrow* pRow;
   VectorPtr<char*>::iterator iName;
   VectorPtr<char*>::iterator iValue;
   S32 iColumn;

   pResultSet = object->GetResultSet(dAtoi(argv[2]));
   if (pResultSet)
   {
      pRow = pResultSet->vRows[pResultSet->iCurrentRow];
      if (!pRow)
         return "invalid_row";

      // Is column specified by a name or an index?
      iColumn = dAtoi(argv[3]);
      if (iColumn == 0)
      {
         // column was specified by a name
         iColumn = object->GetColumnIndex(dAtoi(argv[2]), argv[3]);
         // if this is still 0 then we have some error
         if (iColumn == 0)
            return "invalid_column";
      }

      // We temporarily padded the index in GetColumnIndex() so we could
return a
      // 0 for error.  So now we need to drop it back down.
      iColumn--;
```

```
      // now we should have an index for our column data
      if (pRow->vColumnValues[iColumn])
         return pRow->vColumnValues[iColumn];
      else
         return "NULL";
   }
   else
      return "invalid_result_set";
}

ConsoleMethod(SQLiteObject, escapeString, const char *, 3, 3, "(string)
Escapes the given string, making it safer to pass into a query.")
{
   // essentially what we need to do here is scan the string for any
occurances of: ', ", and \
   // and prepend them with a slash: \', \", \\

   // to do this we first need to know how many characters we are replacing
so we can calculate
   // the size of the new string
   S32 iCount;
   S32 iIndex, iLen, iNewIndex;
   char* szNew;

   iCount = 0;
   iLen = dStrlen(argv[2]);
   for (iIndex = 0; iIndex < iLen; iIndex++)
   {
      if (argv[2][iIndex] == '\'')
         iCount++;
      else if (argv[2][iIndex] == '\"')
         iCount++;
      else if (argv[2][iIndex] == '\\')
         iCount++;

   }
//   Con::printf("escapeString counts %i instances of characters to be
escaped.  New string will be %i characters longer for a total of %i
characters.", iCount, iCount, iLen+iCount);
   szNew = new char[iLen+iCount];
   iNewIndex = 0;
   for (iIndex = 0; iIndex <= iLen; iIndex++)
   {
      if (argv[2][iIndex] == '\'')
      {
         szNew[iNewIndex] = '\\';
         iNewIndex++;
         szNew[iNewIndex] = '\'';
      }
      else if (argv[2][iIndex] == '\"')
      {
         szNew[iNewIndex] = '\\';
         iNewIndex++;
         szNew[iNewIndex] = '\"';
      }
      else if (argv[2][iIndex] == '\\')
      {
```

```
         szNew[iNewIndex] = '\\';
         iNewIndex++;
         szNew[iNewIndex] = '\\';
      }
      else
         szNew[iNewIndex] = argv[2][iIndex];

      iNewIndex++;
   }
//   Con::printf("Last characters of each string (new, old): %s, %s",
argv[2][iIndex-1], szNew[iNewIndex-1]);
//   Con::printf("Old String: %s\nNew String: %s", argv[2], szNew);

   return szNew;
}
```

Appendix 11: Adding and Accessing Data from a SQLite Database

```
function SpecsMenu::onWake(%this)
{
 if($IsClient)
   SpecsMenuWindow.position= EWorldEditor.getMousePosition(); //rest called
by server/client stuff
 else
   {
    if(EWorldEditor.getSelectionSize() $= 0)
    {Canvas.popDialog(SpecsMenu);
        return;
    }
    SpecsMenuWindow.extent= "122 68";
    AutoButton.visible=1;
    //ManualButton.visible=1;
    SpecsButton.setText("Specifications");
    SpecsButton.command="Canvas.pushDialog(SpecsDlg);
Canvas.popDialog(SpecsMenu);";
    if($workDir $= "")

{if(IsFile("../Example/projectOne/data/shapes/furniture/unavailable.jpg"))
//hacktastic  O_o
           $workDir="../Example/";

if(IsFile("../Virt*/projectOne/data/shapes/furniture/unavailable.jpg"))
           $workDir="../Virt*/";
       }
    %temp=(EWorldEditor.getSelectedObject(0)).shapename;
    %j = strstr(%temp, ".");
    %temp = getSubStr(%temp, 0, %j);
    SpecsMenuWindow.position= EWorldEditor.getMousePosition();

    %current =getSubStr(%temp, 0, 999);
    %j = strstr(%current, "/");
    while(%j!=-1)
       {%current=getSubStr(%current, %j+1 , 999);
        %j = strstr(%current, "/");
       }

    %temp = Strreplace(%temp, "/", "_");
    %tempData=(getFieldOfDatabase("SPECS", %temp, 1 ));
    //echo(%tempData);
    //echo(%current);

    if(!(%tempData $= "NULL"))
        Canvas.popDialog(SpecsDlg);
    else
        if((%current $= "red_marker") || (%current $= "blue_marker") ||
(%current $= "green_marker"))
           {SpecsButton.command="Canvas.pushDialog(MarkerDlg);";
            //Canvas.popDialog(SpecsMenu);
            SpecsButton.setText("Attached Comments");
            SpecsMenuWindow.extent= "122 26";
            AutoButton.visible=0;
```

```
            }
        else
            {$idNumOfCurrent= EWorldEditor.getSelectedObject(0);
             SpecsButton.setText("Create Specifications");
             SpecsButton.command="MessageBoxYesNo(\"Specifications\", \"No
specifications exist for this object. Would you like to create some?\",
\"Canvas.pushDialog(SpecsEditDlg);\", \"\");";
             SpecsMenuWindow.extent= "122 26";
             AutoButton.visible=0;
            }
   }
}

function SpecsDlg::onWake(%this)
{
 tab.visible=0; //set to '1' when a menu for loading pictures is created
 //below is a hacky work around to get the pictures of the gui map buttons to
display correctly.
 %pos=button1.position;
 %ext=button1.extent;
 button1.resize(getWord(%pos, 0), getWord(%pos, 1), getWord(%ext, 0),
getWord(%ext, 1));
 %pos=button2.position;
 %ext=button2.extent;
 button2.resize(getWord(%pos, 0), getWord(%pos, 1), getWord(%ext, 0),
getWord(%ext, 1));
 %pos=button3.position;
 %ext=button3.extent;
 button3.resize(getWord(%pos, 0), getWord(%pos, 1), getWord(%ext, 0),
getWord(%ext, 1));
 //end of hack junk
 if($IsClient)
    editSpecsButton.visible=0;
 else
   {
    //default fields before we check the database
    editSpecsButton.visible=0;
    Name.setText("Not Named");
    dimensions.setText("Dimensions: Unknown");
    partnumber.setText("Part Number: Unknown");
    material.setText("Material: Unknown");
    cost.setText("Cost per unit: Unknown");
    picture.setBitmap($workDir @
"projectOne/data/shapes/furniture/unavailable.jpg"); //default is
unavailable.jpg

    //%file = new FileObject();
    $idNumOfCurrent= EWorldEditor.getSelectedObject(0);
    %temp=$idNumOfCurrent.shapename;
    %j = strstr(%temp, ".");
    %temp = getSubStr(%temp, 0, %j);

    if(IsFile($workDir @ %temp @ ".jpg"))
        {picture.setBitmap($workDir @ %temp @ ".jpg");
         tab.visible=0;
        }
```

```
    %temp = Strreplace(%temp, "/", "_");
    %tempData=(getFieldOfDatabase("SPECS", %temp, 1 ));
    if(!(%tempData $= "NULL"))
        {
        //%file.OpenForRead($workDir @ %temp @ ".csv");
        for(%i = 1; %i <= 5; %i++)
          {%line = getFieldOfDatabase("SPECS", %temp, %i);
           if(Strcmp(%line, "")!=0)
             {switch$(%i)
               {case 1:
                    Name.setText(%line);
                case 2:
                   dimensions.setText(%line);
                case 3:
                   partnumber.setText(%line);
                case 4:
                   material.setText(%line);
                case 5:
                   cost.setText(%line);
               }
            }
         }
      //%file.delete();
        }
    else
      Canvas.popDialog(SpecsMenu);
  }
}

function worldEditor::makeMarker(%this, %color)
{
 %pos = ($idNumOfCurrent).getPosition();
 %pos = (getWord(%pos, 0) @ " " @ getWord(%pos, 1) @ " " @ (getWord(%pos,
2)+2.5) );
 //while(IsPointInside(%pos))                 //only check to see if in
interior not inside TSStatic like needed
 //    %pos = (getWord(%pos, 0) @ " " @ getWord(%pos, 1) @ " " @
(getWord(%pos, 2)+1.2) );
 %marker = new TSStatic()
   {
             position = %pos;
             rotation = "1 0 0 0";
             scale = "1 1 1";
             shapeName = "~/data/markers/" @ %color @ ".dts";
             showTerrainInside = "0";
   };
 //echo(%marker);
 %this.clearSelection();
 %this.selectObject(%marker);

 Canvas.pushDialog(markerDlg);
}

function MarkerDlg::onWake(%this)
{if($IsClient)
 {
  markerWindow.setText("Server Marker Information");
```

```
  markerTitle.setText("Marker Info");
  if(GetSubStr(markerName.getText(), 0, 3) $= "red")
     markerPic.setBitmap($workDir @ "projectOne/data/markers/redicon.png");
  else if(GetSubStr(markerName.getText(), 0, 3) $= "blu")
          markerPic.setBitmap($workDir @
"projectOne/data/markers/blueicon.png");
       else if(GetSubStr(markerName.getText(), 0, 3) $= "gre")
               markerPic.setBitmap($workDir @
"projectOne/data/markers/greenicon.png");
 }
 else
 {
    if(!(isFile("MARKERS")))
        CreateDatabase("MARKERS");
    markerComment.setValue("");
    %temp=(EWorldEditor.getSelectedObject(0)).shapename;
    %j = strstr(%temp, ".");
    %temp=getSubStr(%temp, 0 , %j);
    %j = strstr(%temp, "/");
    while(%j!=-1)
        {%temp=getSubStr(%temp, %j+1 , 999);
        %j = strstr(%temp, "/");
        }

    if(Strcmp(%temp, "red_marker")==0)
        markerPic.setBitmap($workDir @
"projectOne/data/markers/redicon.png");
    else if(Strcmp(%temp, "blue_marker")==0)
            markerPic.setBitmap($workDir @
"projectOne/data/markers/blueicon.png");
        else if(Strcmp(%temp, "green_marker")==0)
                markerPic.setBitmap($workDir @
"projectOne/data/markers/greenicon.png");

    if(Strcmp((EWorldEditor.getSelectedObject(0)).getName(), "")==0) //new
marker
        {%i=1;
        while(isObject(%temp @ "_" @ %i))
            %i++;
        markerName.setValue(%temp @ "_" @ %i);
        //echo(%temp @ "_" @ %i);
        }
    else //already has a name so get name and any comments
        {markerName.setValue((EWorldEditor.getSelectedObject(0)).getName());
        //%file = new FileObject();
        //%file.OpenForRead($workDir @ "projectOne/client/Log/" @
(EWorldEditor.getSelectedObject(0)).getName() @ ".csv");
        //echo($workDir @ "projectOne/client/Log/" @
(EWorldEditor.getSelectedObject(0)).getName() @ ".csv");

        %alltext = "";
        %i=1;
        %tempData=(getFieldOfDatabase("MARKERS",
(EWorldEditor.getSelectedObject(0)).getName(), %i ));
        while( !(%tempData $= "NULL"))
          {   %line = getFieldOfDatabase("MARKERS",
(EWorldEditor.getSelectedObject(0)).getName(), %i );
```

```
            %alltext = %alltext @ %line NL "";
            %tempData=(getFieldOfDatabase("MARKERS",
(EWorldEditor.getSelectedObject(0)).getName(), %i++ ));
          }
        markerComment.setValue(%alltext);
        //%file.delete();
        }
 }
}

function MarkerDlg::onSleep(%this)
{
 if(!$IsClient)
   {
    (EWorldEditor.getSelectedObject(0)).setName(markerName.getValue());
    if(!Strcmp((EWorldEditor.getSelectedObject(0)).getName(), "")==0) //only
if has a name
       {
        %file = new FileObject();
        %file.OpenForWrite($workDir @ "projectOne/client/Log/" @
(EWorldEditor.getSelectedObject(0)).getName() @ ".csv");
        clearFieldOfDatabase("MARKERS",
(EWorldEditor.getSelectedObject(0)).getName());
        addFieldToDatabase("MARKERS",
(EWorldEditor.getSelectedObject(0)).getName());
        %temp = getSubStr(markerComment.getValue(), 0, 9999);
        %j = 0;

        while(%j!=-1)
          {
           %j = strstr(%temp, "\n");
           if(%j == -1)
              %j = 9999;
           %outline = getSubStr(%temp, 0, %j);
           %temp = getSubStr(%temp, %j+1, 9999);
           if(Strcmp(%outline, "")!=0)
               {appendFieldOfDatabase("MARKERS", %outline,
(EWorldEditor.getSelectedObject(0)).getName());
                %file.writeLine(%outline);
               }
           if(%j == 9999)
              %j = -1;
          }
        %file.delete();
       }
   }
}

function SpecsEditDlg::onWake(%this)
{
 %file = new FileObject();
 %temp=$idNumOfCurrent.shapename;
 %j = strstr(%temp, ".");
 %temp = getSubStr(%temp, 0, %j);
 %temp = Strreplace(%temp, "/", "_");

 %tempData=(getFieldOfDatabase("SPECS", %temp, 1 ));
```

```
 if(!(%tempData $= "NULL"))
    {
     //%file.OpenForRead($workDir @ %temp @ ".csv");
     //desc.setValue(%file.readline());
     //dim.setValue(getSubStr(%file.readline(), 12, 999));  // dont display
'field: '
     //part.setValue(getSubStr(%file.readline(), 13, 999));
     //mat.setValue(getSubStr(%file.readline(), 10, 999));
     //cpu.setValue(getSubStr(%file.readline(), 15, 999));
     desc.setValue(getFieldOfDatabase("SPECS", %temp, 1));
     dim.setValue(getSubStr(getFieldOfDatabase("SPECS", %temp, 2), 12, 999));
// dont display 'field: '
     part.setValue(getSubStr(getFieldOfDatabase("SPECS", %temp, 3), 13,
999));
     mat.setValue(getSubStr(getFieldOfDatabase("SPECS", %temp, 4), 10, 999));
     cpu.setValue(getSubStr(getFieldOfDatabase("SPECS", %temp, 5), 15, 999));
    }
 //%file.delete();
}

function SpecsEditDlg::onSleep(%this)
{EWorldEditor.clearSelection();
 EWorldEditor.selectObject($idNumOfCurrent);
 desc.setValue("");
 dim.setValue("");
 part.setValue("");
 mat.setValue("");
 cpu.setValue("");
}

function SpecsEditDlg::onSave(%this)
{if(!(isFile("SPECS")))
    CreateDatabase("SPECS");
 %file = new FileObject();
 %temp=$idNumOfCurrent.shapename;
 %j = strstr(%temp, ".");
 %temp = getSubStr(%temp, 0, %j);

 %file.OpenForWrite($workDir @ %temp @ ".csv");
 %file.writeline(Strreplace(desc.getValue(), ",", " "));//no commas!
 %file.writeline("Dimensions: " @ Strreplace(dim.getValue(), ",", " "));
 %file.writeline("Part Number: " @ Strreplace(part.getValue(), ",", " "));
 %file.writeline("Material: " @ Strreplace(mat.getValue(), ",", " "));
 %file.writeline("Cost per unit: " @ Strreplace(cpu.getValue(), ",", " "));
 %file.delete();

 %temp = Strreplace(%temp, "/", "_");
 clearFieldOfDatabase("SPECS", %temp);
 addFieldToDatabase("SPECS", %temp);
 appendFieldOfDatabase("SPECS", desc.getValue(), %temp);
 appendFieldOfDatabase("SPECS", "Dimensions: " @ dim.getValue(), %temp);
 appendFieldOfDatabase("SPECS", "Part Number: " @ part.getValue(), %temp);
 appendFieldOfDatabase("SPECS", "Material: " @ mat.getValue(), %temp);
 appendFieldOfDatabase("SPECS", "Cost per unit: " @ cpu.getValue(), %temp);
}
```